(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) EP 1 304 871 A2

(12) EUROPEAN PATENT APPLICATION

(43) Date of publication:
23.04.2003 Bulletin 2003/17

(51) Int Cl.7: H04N 5/00, H04N 7/24,
H04N 7/167

(21) Application number: 02253866.4

(22) Date of filing: 31.05.2002

(84) Designated Contracting States:
AT BE CH CY DE DK ES FI FR GB GR IE IT LI LU MC NL PT SE TR
Designated Extension States:
AL LT LV MK RO SI

(30) Priority: 21.08.2001 EP 01402202

(71) Applicant: Canal+ Technologies Société Anonyme
75906 Paris Cedex 15 (FR)

(72) Inventors:
- Letourneur, Patrice
c/o Canal+ Technologies S.A.
75906 Paris, Cedex 15 (FR)
- Le Crom, François c/o Canal+ Technologies S.A.
75906 Paris, Cedex 15 (FR)
- Maillard, Michel c/o Canal+ Technologies S.A.
75906 Paris, Cedex 15 (FR)

(74) Representative: Rinuy, Santarelli
14, avenue de la Grande Armée
75017 Paris (FR)

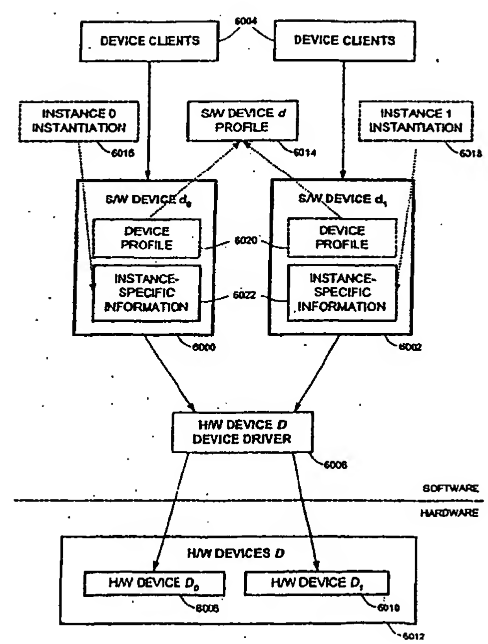
Remarks:
Claims 19-85 are deemed to be abandoned due to non-payment of the claims fees (Rule 31 (2) EPC).

(54) Method and apparatus for a receiver/decoder

(57) A method and apparatus relating to a receiver/decoder in a digital television environment, including logical devices (including logical demultiplexer devices) for representing physical and other devices in the receiver/decoder. The method includes the instantiation of devices by the receiver/decoder as required to support functionality thereof. The method further includes the use of multiple demultiplexers/remultiplexers, for example in the recording of more than one service simultaneously; and a control word device for the management of control word operations; the use of two or more tuners.

Various elements of a digital television system (such as a receiver/decoder and a set top box) are also disclosed.

Fig. 9



EP 1 304 871 A2

Printed by Jouve, 75001 PARIS (FR)

each identifier being for use in controlling the descrambling equipment with respect to a respective instance of use of the descrambling equipment.

[0065] At least first and second instances of use of the descrambling equipment may be subject to at least one different respective access condition for descrambling signals.

[0066] The apparatus may be arranged to control supply of a decryption key to the descrambling equipment for use in a respective descrambling process. For example, the apparatus may be arranged to co-ordinate supply of a decryption key, from means for receiving delivery of decryption keys, to the descrambling equipment, the co-ordination being done by use of the identifier assigned to each respective instance of use of the descrambling equipment. The decryption key may, for example, be received in a multiplexed information signal.

[0067] The interface may be accessible to at least one device in a family of devices supporting a demultiplexing function and the apparatus may further comprise means to provide more than one instance of the same device in the family of devices, each instance so provided being related to at least one assigned identifier for an instance of use of the descrambling equipment.

[0068] In yet a further aspect of the invention, there is provided a method of descrambling scrambled signals, comprising the use of an interface in the control of descrambling equipment. The method may, for example, comprise the steps of assigning an identifier to an instance of use of the descrambling equipment; and using the identifier in controlling the descrambling equipment in relation to said instance of use by means of the interface.

[0069] The method may further comprise assigning a first identifier to a first instance of use of the descrambling equipment, assigning a second identifier to a second instance of use of the descrambling equipment, and using the first and second identifiers to distinguish the first and second instances of use in controlling the descrambling equipment, said first and second instances of use occurring over a common time period.

[0070] The first and second instances of use of the descrambling equipment may be subject to at least one different respective access condition for descrambling.

[0071] In a further aspect of the invention, there is provided apparatus for processing data, comprising means for recording a first service, means for simultaneously recording a second service and means for playing back the first service and the second service at respective times chosen by a user. Such apparatus may provide increased flexibility and utility for the user.

[0072] In yet a further aspect, the invention provides apparatus for a receiver/decoder comprising recording means for recording two or more programme data streams over a common time period.

[0073] In yet a further aspect, the invention provides a method of recording programme signals, comprising recording two or more different programme data streams over a common time period.

[0074] In a further aspect of the present invention, there is provided apparatus for processing data, comprising means for receiving service data on a first channel and means for receiving conditional access data relating to that service on a second channel. This may confer the advantage of reducing the bandwidth required to receive conditional access.

[0075] The apparatus may further comprise means for causing the conditional access data receiving means to change the channel from which it receives the conditional access data. It may thus be possible to receive conditional access data from a channel which changes, for example periodically, enabling the channel upon which conditional access data is broadcast to change, thus allowing channel-hopping of conditional access data, which may increase security.

[0076] To this end, a further aspect of the invention provides a broadcast centre comprising service data broadcasting means adapted to broadcast service data excluding conditional access data on a first channel, and conditional access data broadcasting means for broadcasting conditional access data relating to the service on a second channel.

[0077] The broadcast centre may further comprise means for causing the conditional access data broadcasting means to change the channel upon which the conditional access data is broadcast.

[0078] In yet a further aspect of the invention, there is provided a conditional access system comprising service data broadcasting means for broadcasting service data excluding conditional access data on a first channel, conditional access data broadcasting means for broadcasting conditional access data relating to the service on a second channel, service data receiving means for receiving the service data and conditional access data receiving means for receiving the conditional access data.

[0079] In a further aspect, the invention provides apparatus for a receiver/decoder, for use in receiving and/or decoding signals received at the apparatus over more than one channel, wherein said apparatus comprises at least two inputs for connection to respective channels and correlation means for correlating a signal received at a first of said inputs with a signal received at a second of said inputs.

[0080] The apparatus may further comprise detection means for detecting a channel identifier received in the signal at the first input, channel selection means for selecting a channel for connection to said second input, and control means to control the channel selection means to select a channel identified by the channel identifier for connection to said second input. Each respective channel may for example have a different carrier frequency.

[0081] The signal received at the first input may comprise at least primarily content data and the signal re-

ceived at the second input may comprise administrative data with respect to the content data.

[0082] The apparatus may further comprise an interface for use in controlling descrambling equipment to descramble information signals.

[0083] In a further aspect of the invention, there is provided a broadcast system for broadcasting related signals in multiplexed signal transport streams, which comprises broadcast means for broadcasting a first signal in a first multiplexed signal transport stream, broadcast means for broadcasting a second signal in a second multiplexed signal transport stream, and correlation signal transmission means for transmitting a correlation signal for use in correlating the related signals. The correlation signal may for example comprise carrier frequency data for identifying a carrier frequency for at least one of the first and second transport streams. The correlation signal may additionally or alternatively comprise at least one slot identifier for identifying a slot in one of the first and second transport streams.

[0084] The invention also provides a computer program and a computer program product for carrying out any of the methods described herein and/or for embodying any of the apparatus features described herein, and a computer readable medium having stored thereon a program for carrying out any of the methods described herein and/or for embodying any of the apparatus features described herein.

[0085] The invention also provides a signal embodying a computer program for carrying out any of the methods described herein and/or for embodying any of the apparatus features described herein, a method of transmitting such a signal, and a computer product having an operating system which supports a computer program for carrying out any of the methods described herein and/or for embodying any of the apparatus features described herein.

[0086] The invention extends to methods and/or apparatus substantially as herein described with reference to the accompanying drawings.

[0087] Any feature in one aspect of the invention may be applied to other aspects of the invention, in any appropriate combination. In particular, method aspects may be applied to apparatus aspects, and vice versa.

[0088] Furthermore, features implemented in hardware may generally be implemented in software, and vice versa. Any reference to software and hardware features herein should be construed accordingly.

[0089] Preferred features of the present invention will now be described, purely by way of example, with reference to the accompanying drawings, in which:-

Figure 1 is an overview of a satellite digital television system;

Figure 2 is an overview of a cable digital television system;

Figure 3 is an overall system view, with the head-end shown in more detail;

Figure 4 is a schematic of the component architecture of the receiver/decoder;

Figure 5 is a diagram of the software architecture of the receiver/decoder;

Figure 6 is a diagram showing the top half of Figure 5 in more detail;

Figure 7 is a diagram showing the bottom half of Figure 5 in more detail;

Figure 8 is a diagram showing an alternative embodiment of the bottom half of Figure 5;

Figure 9 shows the structure of software devices in accordance with an embodiment;

Figure 10 shows the structure of a compound device identifier;

Figures 11a and b represent respectively a class description for a device class and a set of instantiation data for a device instance;

Figure 12 shows the three types of logical demultiplexer and their supporting devices in accordance with an embodiment;

Figures 13a, b and c represent stages in a process for selecting a hardware demultiplexer;

Figure 14 is a flow diagram for the process to which Figure 13 relates;

Figure 15 shows time lines illustrating the facility of an embodiment for recording more than one service simultaneously;

Figure 16 illustrates the conditional access operations performed by an embodiment when demultiplexing and descrambling a service; and

Figure 17 is an overview of a system for providing conditional access data.

System overview

[0090] An overview of a digital television system 500 is shown in Figure 1. As will be discussed below, the system 500 comprises a broadcast centre 1000, a receiver/decoder 2000, a software/hardware architecture 3000 of the receiver/decoder, an interactive system 4000, and a conditional access system 5000, as will all be discussed below.

[0091] The system 500 includes a mostly conventional digital television system 502 that uses the known MPEG-2 compression system to transmit compressed digital signals. In more detail, MPEG-2 compressor 1010 in a broadcast centre 1000 receives a digital signal stream (typically a stream of video signals). The compressor 1010 is connected by linkage 1020 to a multiplexer and scrambler 1030.

[0092] The multiplexer 1030 receives a plurality of further input signals, assembles the transport stream and transmits compressed digital signals to a transmitter 510 of the broadcast centre via linkage 1022, which can of course take a wide variety of forms including telecommunications links. The transmitter 510 transmits electromagnetic signals via uplink 514 towards a satellite transponder 520, where they are electronically proc-

tions to be read. These include Java interpreters 3512, PanTalk interpreters 3514, HTML interpreters 3516, MHEG-5 interpreters 3518 and others.

- Service Information (SI) Engine. The SI Engine 3540 loads and monitors common Digital Video Broadcasting (DVB) or Program System Information Protocol (PSIP) tables and puts them into a cache. It allows access to these tables by applications which need the data contained in them.
- Scheduler 3542. This module allows for pre-emptive, multithreaded scheduling with each thread having its own event queue.
- Memory Manager 3544. This module manages the access to memory. It also automatically compresses data in memory when necessary and performs automatic garbage collection.
- Event Manager 3546. This module allows events to be triggered according to priority. It manages timer and event grabbing and allows applications to send events to each other.
- Dynamic Linker 3548. This module allows the resolution of addresses arising from native Java functions, loads native methods from a Java class downloaded into RAM and resolves calls from downloaded native codes towards ROM.
- Graphics System 3550. This system is object-orientated and optimized. It includes graphic window and object management as well as a vectorial font engine with multi-language support.
- Class Manager 3552. This module loads classes and resolves any class referencing problems.
- File System 3554. This module is compact and optimized to manage a hierarchical file system with multiple ROM, flash, RAM and DSMCC volumes. Flash integrity is guaranteed against any incidents.
- Security Manager 3556. This module authenticates applications and controls the access of applications to sensitive memory and other zones of the set-top box.
- Downloader 3558. This module uses automatic data loading from a remote DSMCC carousel or through the NFS protocol, with downloaded files accessed in the same way as resident ones. Memory clear-up, compression and authentication are also provided.

[0141] Furthermore, the DAVIC resource notification model is supported so that client resources are efficiently managed.

[0142] A kernel 3650 manages the various different processes running in the virtual machine 3500 and device layer interface 3700 (not shown). For efficiency and reliability reasons, the kernel implements relevant parts of the POSIX standard for operating systems.

[0143] Under control of the kernel, the virtual machine (running Java and Pantalk applications) runs in its own thread, separate to other 'server' elements of the operating system, such as the mass storage server 3850 (not shown). Corresponding provisions, such as requiring Thread IDs to be passed as parameters in system calls, are also made in the API layer 3300 to allow the applications 3120 to benefit from the multithreaded environment.

[0144] By providing multiple threads, more stability can be achieved. For example, if the virtual machine 3500 ceases to operate for some reason, by suffering a crash or being blocked for a long time by an application trying to access a device, other time-critical parts of the system, such as the hard disk server, can continue to operate.

[0145] As well as the virtual machine 3500 and kernel 3650, a hard disk video recorder (HDVR) module 3850 is provided for handling the recording and playback functions of the hard disk 2210 or other attached mass storage component. The server comprises two separate threads 3854, 3856 handling recording, one thread 3858 for handling playback, and a file system library 3852 for interfacing with the mass storage components.

[0146] An appropriate one of the threads 3854, 3856, 3858 in the hard disk video recorder (HDVR) 3850 receives commands (such as a command to start recording a particular programme) from clients such as the personal video recorder (PVR) application 3154, in response to the user pressing a 'record' button, for example.

[0147] In turn, the thread in question then interacts with the service device 3736 (shown in Figure 7) to set up and synchronise the parts of the receiver/decoder handling the bitstream to be recorded or played back. In parallel, the thread also interacts with the file system library 3852 to coordinate the recording or playback operation at appropriate places on the hard disk 2210 (not shown).

[0148] The file system library 3852 then sends commands to the mass storage device 3728 (also shown in Figure 7) which tell the mass storage device 3728 which sub-transport stream (STS) to transfer (via a FIFO buffer), and on which hard disk target the stream should be stored. Allocation of clusters on the hard disk and general file management is carried out by the file system library 3852, the mass storage device itself being concerned with lower level operations.

[0149] The service device 3736 mentioned above is unique amongst the devices in that it does not relate to a physical component of the receiver/decoder. It instead provides a high level interface which groups together in a single 'instance' the various sets of tuner, demultiplexer, remultiplexer and hard disk devices in the receiver/decoder, freeing higher level processes from the difficulties of coordinating the various sub-devices.

[0150] With reference to Figure 7 the software architecture of the receiver/decoder 3000 corresponding to the bottom half of Figure 5 (comprising the device layer interface 3700 and the system software and hardware layer 3900) will now be described in more detail.

[0151] Further devices provided in the device layer in-

clude the conditional access device 3720, tuner devices 3724 corresponding to the two (or potentially more) tuners 2016, 2018 of Figure 4, the video device 3734, the I/O port device 3726, and the service device 3736 and mass storage device 3728 mentioned above.

[0152] In broad terms, a device can be regarded as defining a logical interface, so that two different devices may be coupled to a common physical port. Certain devices may communicate among themselves, and all devices also operate under the control of the kemel 3650.

[0153] Before using the services of any device, a program (such as an application instruction sequence) has to be declared as a "client", that is, a logical access-way to the device or the device manager 3710. The manager gives the client a client number which is referred to in all accesses to the device. A device can have several clients, the number of clients for each device being specified depending on the type of device. A client is introduced to the device by a procedure "Device: Open Channel". This procedure assigns a client number to the client. A client can be taken out of the device manager 3710 client list by a procedure "Device: Close Channel".

[0154] The access to devices provided by the device manager 3710 can be either synchronous or asynchronous. For synchronous access, a procedure "Device: Call" is used. This is a means of accessing data which is immediately available or a functionality which does not involve waiting for the desired response. For asynchronous access, a procedure "Device: I/O" is used. This is a means of accessing data which involves waiting for a response, for example scanning tuner frequencies to find a multiplex or getting back a table from the MPEG stream. When the requested result is available, an event is put in the queue of the engine to signal its arrival. A further procedure "Device: Event" provides a means of managing unexpected events.

[0155] In a second embodiment of the receiver/decoder, the lower half of the architecture of the receiver/decoder is replaced by the layers shown in Figure 8.

[0156] In this embodiment, an extended device layer interface (EDLI) 3600 is provided between the virtual machine 3500 (not shown) and the device layer interface 3700, and an abstraction device interface 3800 is provided between the device layer interface 3700 and the system software/hardware layer 3900. Otherwise, like parts are indicated with like reference numerals.

[0157] The extended device layer interface (EDLI) 3600 provides a dedicated interface between the virtual machine 3500 and the device layer interface 3700 and generally provides multithreading support to the device layer interface. Functions of the EDLI include routing asynchronous events to the appropriate thread in the middleware (since the device layer interface need not itself support multithreading) and routing messages between threads.

[0158] The abstraction device interface 3800 provides a further interface between the device layer interface 3700 and the device drivers 3910 in the system software/hardware layer 3900. By providing such an interface, the large and complex device layer 3700 can be made hardware independent to a greater degree.

Further aspects of system devices

[0159] The organisation of software devices within the receiver/decoder 2000, and in particular the use of device instantiation and logical devices to provide enhanced functionality, are described in more detail below. A logical demultiplexer device (otherwise referred to as a 'DEMUX device'), which advantageously makes use of the above-mentioned features of device instantiation and logical devices, is then described.

[0160] Subsequently, the use of the above-mentioned DEMUX device for demultiplexing more than one service simultaneously (to allow one demultiplexer to perform the role of two conventional demultiplexers, for example) and for recording more than one service (such as a digital television programme) simultaneously are then described. The provision of a control word device and other system aspects for use in managing conditional access will then be described, and finally there follows description of the use of two tuners, particularly with respect to conditional access data and having a close relationship with the above-mentioned DEMUX device(s).

Device instantiation in the context of device management

[0161] In the preferred embodiment, the device manager 3710 is adapted to instantiate the devices required by the receiver/decoder. This instantiation of software devices and their detailed structure is now described in more detail, with reference to Figure 9.

[0162] Figure 9 shows two software devices d_0 6000 and d_1 6002, their respective clients 6004, a device driver 6006 for the corresponding hardware device class *D*, and the corresponding hardware devices D_0 6008 and D_1 6010, which form part of the same class 6012 of devices (of type *D*). Also shown is the (software) device profile *d* 6014, and the instantiation data 6016, 6018 for each respective device 6000, 6002. As will be described in more detail below, the devices 6000, 6002 themselves comprise a portion 6020 corresponding to the device profile, and a portion 6022 corresponding to instance-specific information. It should be noted that the principles discussed below in relation to Figure 9 apply also to any number of software and hardware devices other than the two shown.

[0163] The devices 6000, 6002 form part of the device layer 3700; the device clients typically form part of the application layer 3100, API layer 3300, virtual machine layer 3500 or device layer 3700; and the device driver 6006 and hardware devices 6008, 6010 form part of the system software/hardware layer 3900, all as shown in Figures 6, 8 and 9.

[0164] In order to create greater flexibility and efficiency, each of the 'software devices' referred to elsewhere is created by a process of instantiation. In more detail, when the receiver/decoder 2000 is initialised or reinitialised, the device manager 3710 instantiates in turn each of the software devices — such as devices d_0 6000 and d_1 6002 — by combining a general 'device profile' for each required device — such as the device profile *d* 6014 — with instance-specific information (which may be no more than the instance identifier) — such as instantiation information 6016 or 6018.

[0165] In the preferred embodiment, the device profile comprises a stand-alone version of the corresponding instantiated device, including the necessary interfaces, code and data fields, and the process of instantiation involves creating a byte-for-byte copy of the device profile and filling in the relevant data fields with the instance-specific information. Whilst this results in some code duplication, every instance of a particular device is then entirely independent from other instances of the same or other devices.

[0166] In variants of the preferred embodiment, the device profile again comprises the necessary interfaces and code generic to the device type, but the device instance only comprises the instance-specific data, with a set of pointers or other references to the parent profile. In these and other variants, calls made through the device manager to a specified device instance are routed by the device manager to the parent device profile (since the device instances in these cases contain only data), with the device manager also providing to the device profile functions the appropriate instance information, typically in the form of a pointer to the data.

[0167] It should also be noted that in Figure 9 a generic device driver 6006 is provided for accessing the hardware functions of the devices *D*; in this case, calls to this device driver (as opposed to higher level calls to the device manager 3710) specify the particular one of the hardware devices to which the call relates (by supplying the device instance number as a parameter, for example). Depending on the implementation of the device layer 3700 and system software/hardware layer 3900, which can vary widely, separate device drivers might instead be provided for each of the hardware devices 6008, 6010.

[0168] Each class or type of device is allocated a two-byte class identifier 6102 which identifies that class of device uniquely. Upon instantiation, each device of a particular class is allocated a two-byte instance identifier 6104 which is unique within that class. That is to say the first TUNER device may be allocated the same instance identifier as the first DEMUX device (described below); however, no two TUNER devices will have the same instance identifier. After instantiation, each device is uniquely identified by a compound, four-byte identifier 6100 (see Figure 10) constructed from the class identifier 6102 (forming the lowest two bytes of the compound identifier) and the instance identifier 6104 (forming the highest two bytes of the identifier).

[0169] In more detail, the devices *D* could be the tuners in the receiver/decoder, for example, which, in broad terms, are adapted to select for decoding a number of different transport streams broadcast on different frequencies.

[0170] In this case, the hardware devices D_0 and D_1 would be the two tuners 2016, 2018 shown in Figure 4 respectively, the software devices d_0 6000 and d_1 6002 would be the two TUNER devices 3724 shown in Figures 7 and 8, and the hardware device driver would be a tuner device driver (not shown elsewhere), which would allow direct communication with the hardware underlying the two tuners (although as noted above, separate device drivers could be provided for each tuner).

[0171] To illustrate the above, a typical function provided by the generic TUNER device (*d*) is the TUNER_TUNING_SET command, for example, which sets various properties of the given tuner, including frequency, signal polarisation, and so on. The TUNER_TUNING_SET command is used, for example, by the zapping application 3146 mentioned earlier in order to change channels. In this system, the actual executable code corresponding to the TUNER_TUNING_SET command is provided in the device profile, and is common to all TUNER devices. Information such as the current frequency of a particular tuner instance (such as d_0) is contained in the instance-specific information for the particular TUNER device (d_0).

[0172] The device profile 6014 and the instantiation data 6016, 6018 will now be further described with reference to Figures 11a and 11b.

[0173] Figure 11a shows a class description or device profile 6014 for a device which comprises in a first portion 6014a the interfaces, code and data fields necessary for instances of the device class, and two additional data fields 6014b, 6014c. The first additional field 6014b contains the device identifier 6102; the second 6014c contains a value for a maximum number of instances of that class of device. The maximum number of instances of a particular software device and the maximum total number of software device instances is determined by constraints of the system software (for example the size of the device ID described above), rather than the number of underlying hardware devices present in the system.

[0174] Referring to Figure 11b, upon initialisation of the receiver/decoder, the physical devices present in the system become known to the device layer interface, the required software devices are instantiated and provided with instantiation data 6016. The instantiation data includes the compound four-byte device identifier described above. The instantiation data further comprises default values and/or possible values or ranges of values for procedure parameters, which are determined upon initialisation of the receiver/decoder by polling the system hardware.

[0175] In contrast to the situation shown in Figure 9, where there are a plurality of hardware devices and corresponding software devices (such as the two tuners and the corresponding TUNER software devices mentioned above), there may be only one software device in a given device class (such as for the LCARD smartcard reader device, for example). In this case, a device profile is still provided, from which a single instance is created, as described above.

[0176] The stored operating system comprises the device profiles necessary to create all of the required software device instances. As described above, in the preferred embodiment, when the operating system is booted or rebooted (following a power-on or other reset of the receiver/decoder), a special portion of the operating system first scans the hardware to detect the attached hardware devices, and then creates all of the appropriate software device instances from the stored device profiles, before the (re)boot procedure continues. Thus, the instantiation of software devices is static, performed once every reset. Since the hardware provided in the receiver/decoder is also static, this is generally perfectly acceptable.

[0177] In variants of the preferred embodiment, for example where components of the receiver/decoder are 'hot-swapped', the instantiation of devices is dynamic— that is, on demand. In these variants, functions are provided as appropriate by the device manager to create and delete instances.

[0178] In the preferred embodiment, different portions of the middleware of the receiver/decoder responsible for the various devices (such as the HDVR module, for example, which is at least in part responsible for the mass storage device) add themselves to a list (maintained by the operating system) of processes to be called when the devices are to be created. In more detail, to create all of the instances, the operating system calls each of the processes on the list in turn, so each process itself creates the necessary device instances (using central functions provided by the device manager) and sets the instance-specific information accordingly.

[0179] By having each process in charge of given devices setting the respective instance-specific data, the devices can be easily customised.

[0180] In some cases, as will be described in more detail later, a particular functionality offered by a subset of a hardware device or a group of hardware devices is encapsulated in a single software device, providing a 'logical' software device without a physical equivalent. An example of this would be the demultiplexer device ('DEMUX' device), which is described later.

[0181] The communication of devices with one another and with applications under the control of the device manager 3710 is now described in more detail.

[0182] Each device communicates with an application, or potentially another device, under the control of the device manager 3710. The device manager 3710 controls access to the devices, declares receipt of an unexpected event and manages shared memory. The three procedures used by the devices for communication with one another and with other parts of the system (for example, applications) are introduced above. In greater detail, these three procedures serve the following purposes:

[0183] "Device: Call" procedures are used to give synchronous commands or effect data transfer. Execution of an application requesting this procedure is suspended during execution of the command or the data transfer. This allows operations which must be performed in strict sequence to be controlled reliably.

[0184] "Device: I/O" procedures provide for asynchronous operation. That is, an application can send a request for a data transfer or a particular function to be performed by a device, and execution of the requesting application continues while the data transfer or function is performed. When a requested result is available, an event is put in a queue by the device manager 3710 to signal its arrival.

[0185] "Device: Event" procedures provide a means of managing unexpected events, enabling events to be signalled to an application by a device. When a device makes such a call, the device manager 3710 loads an event item into a queue. The Event Manager 3546 of the virtual machine 3500 extracts event items according to a priority level allocated to each event item and inserts them into appropriate further queue structures specific to particular applications in the virtual machine 3500. When an application receives an event, it is interrupted and responds independently of the code it was executing at the time an event is signalled. Events may be used to notify something which has occurred on an interface, such as a bus reset, or can be used to notify a result in response to an asynchronous command for instance to signal completion of a requested data transfer.

[0186] As a consequence of the device management described above, full device instance specific information is not required at compile-time; information necessary in use of the receiver/decoder can be made available subsequently, when the device layer interface 3700 is installed together with a platform such as the system software/hardware 3900.

[0187] If a particular function of a class of software device is not supported by the underlying hardware (that is to say that, if a device feature, such as a scan feature for a tuner device for example, is not provided in hardware), instances of that software device will not provide that function. If none of the functions provided by a class of software device are supported by the underlying hardware, no instances of that device will be instantiated.

[0188] As indicated earlier, an important feature is the abstraction of functionality from its supporting physical device(s). Each function can potentially be a complex one, involving several physical devices, and a set of functions may be related in that they involve an overlapping set of physical or logical devices. This is dealt with

by device instantiation as described generally above. One device class can be instantiated with respect to more than one function. This is exemplified by the description of multiple demultiplexing functions in a receiver/decoder below, each demultiplexing function having its own set of device instances and the device instances belonging to overlapping sets of device classes.

Logical devices

[0189] As indicated above, the preferred embodiment comprises logical devices. A logical device is one which may, but does not necessarily, correspond to a single item of underlying hardware. The functionality provided by a logical device may for example be a subset of the functionality provided by an item of hardware or it may be an amalgamation of functionality provided by a number of items of hardware. It may include functionality provided by the software device itself without the support of an item or items of hardware. Furthermore, a logical device may provide functionality which it accesses by making calls to one or more further software devices, which may provide that functionality with or without the support of an item or items of hardware.

[0190] The preferred embodiment provides demultiplexing functionality to applications using logical demultiplexer (DEMUX) devices, which make calls both to hardware and to other software devices.

[0191] A set of software devices is provided in the device layer interface 3700 for accessing demultiplexing functions of the DDR unit 2010 in response to a request by an application. This set comprises:

- a DEMUX (logical demultiplexer) device (3762 in Figure 12) for extracting packets from a transport stream and for coordinating the activities of the other devices;
- an MLOAD device 3738 for extracting data sections from packets of a transport stream,
- an MCOM device 3764 for transfer of data sections from an transport stream to a communication port, including the filtering and manipulation of that data,
- a CA device 3720 for overseeing conditional access operations,
- a CW device 3760 (further described below) for passing control words to descramblers,
- a TS_REMUX device 3766 for assembling a transport stream, and
- a SERVICE device 3736 for overseeing the presentation of a service to a viewer.

[0192] In the preferred embodiment, three types of logical demultiplexer device are provided (see Figure 12):

- PLAYER 6220, which is able to play a service (including the extraction of packets from a transport stream, descrambling those packets and routing them to the correct part of the system for display),
- RECORDER 6230, which is able to record a service (including the extraction of packets from a transport stream, the construction of a programme stream and the routing of that stream to the correct part of the system for recording), and
- PLAYER/RECORDER 6240, which is capable of performing the actions of both of the above types of demultiplexer device.

[0193] These three types of logical demultiplexer are supported by different subsets 6222, 6232, 6242 of the software devices listed above, as can be seen in Figure 12. The PLAYER logical demultiplexer 6220 does not require a TS_REMUX device 3766 since it does not need to construct a programme transport stream, and the RECORDER logical demultiplexer 6232 does not require a SERVICE device 3736 since it is not concerned with the presentation of a service to a viewer.

[0194] In the preferred embodiment, one of each of the types of logical demultiplexer is instantiated; and for each logical demultiplexer a dedicated one of each of the required other software devices listed above is also instantiated.

[0195] When a logical demultiplexer requires functionality provided by a hardware demultiplexer, for example to record a service from an incoming transport stream, the logical demultiplexer first notifies the hardware demultiplexer of the source of the transport stream using a DEMUX_SET_SOURCE command. The possible sources include one of the tuners (further information regarding the provision of multiple tuners is provided below), the hard disk via a buffer (for example, a FIFO), and a port to which an external device is attached. If the demultiplexer is active in demultiplexing a stream from a different source, it notifies the logical demultiplexer device that it is not available to perform the requested operation. Otherwise the source of the demultiplexer is set to the indicated source and the requested operation is performed. The procedure for allocating a hardware demultiplexer to perform a desired demultiplexing operation is further described below.

[0196] In a particularly preferred embodiment, the logical devices comprise one or more identifier identifying one or more physical or other software devices which implement the functionality offered by the logical device. In a further preferred embodiment, a database containing data indicating which logical devices uses which other device(s) is maintained, for example by the device manager.

Multiple demultiplexers/remultiplexers

[0197] In a preferred embodiment, the DDR 2010 comprises two (or, in a particularly preferred embodiment, three) physical demultiplexers, each capable of being set to demultiplex up to 32 PIDs received from a distinct source. Moreover, preferred embodiments are

able, for example using the logical demultiplexer devices described above, to use each physical demultiplexer to demultiplex more than one service from a particular source simultaneously.

[0198] In order illustrate this feature, the procedure for the selection of a physical demultiplexer to perform a desired demultiplexing operation will now be described, with reference to Figures 13a, b and c, and 14, in the context of a receiver/decoder having two physical demultiplexers 6302, 6304.

[0199] Figures 13a and b show first and second tuners 2016 and 2018 and (schematically) first and second demultiplexers 6302 and 6304. In Figure 13a, the first demultiplexer 6302 is active in demultiplexing a service being received in a transport stream X 6310 to which the first tuner 2016 is tuned; the first tuner 2016 is therefore set as the source 6250 of the first demultiplexer. The following describes the procedure undertaken when, under these circumstances, it is desired to demultiplex a service comprising 5 PIDs from a second transport stream Y 6320.

[0200] A logical demultiplexer device (as described above) is allocated to oversee the demultiplexing operation. The logical demultiplexer then performs the following procedure, described with reference to Figure 14:

[0201] The logical demultiplexer device determines (at step 7002) whether any physical demultiplexer is demultiplexing a service from the desired transport stream. If so, then it checks (at step 7004) whether that physical demultiplexer has sufficient capacity to demultiplex five additional PIDs. If the answer is yes again, the logical demultiplexer uses that physical demultiplexer to perform the desired operation.

[0202] In the case of this example, however, no physical demultiplexer has the correct source for the desired operation (that is, a tuner tuned to transport stream Y). Therefore, after step 7002, the logical demultiplexer checks (at step 7006) whether any physical demultiplexer is inactive by determining whether any has its source disconnected. If no physical demultiplexers are inactive, then the logical demultiplexer returns an error (at step 7008) to the application which requested the demultiplexing operation. However, in the case of this example, the source of the second physical demultiplexer is disconnected, and it is therefore available to be used by the logical demultiplexer, which switches the source of the second physical demultiplexer to the second tuner 2018, and requests that the second tuner be tuned to the frequency of the desired transport stream (in the preferred embodiment, by sending a command to a TUNER device 3724 as described above).

[0203] In a second example, commencing with the situation shown in Figure 13a, it is desired to demultiplex five additional PIDs from transport stream X 6310. In this case, at step 7002, it is discovered that the first physical demultiplexer 6302 does have the desired source. The logical demultiplexer then determines (at step 7004) whether the first physical demultiplexer 6302 has sufficient capacity to demultiplex five additional PIDs; it does since it is presently demultiplexing only five PIDs (as stated above, each demultiplexer may simultaneously demultiplex 32 PIDs from a transport stream). The logical demultiplexer therefore uses the first physical demultiplexer 6302 to perform the desired operation, switching its source to the first tuner 2016, as shown in Figure 13c.

[0204] The preferred embodiment also comprises two (or more) physical remultiplexers, each capable of constructing, in a known manner, a transport stream for storage from the demultiplexed packets of a service

[0205] The embodiment is thus capable of presenting more than one service simultaneously (for example, in a picture-in-picture configuration) or recording more than one service simultaneously (upon which subject more Information is provided below), or recording one or more services while presenting one or more different services, regardless of whether the services being presented/recorded are being received on the same or different transport streams.

Recording multiple services

[0206] As indicated above, the preferred embodiment is capable of recording more than one service simultaneously.

[0207] The requirement to do so may arise, for example, when a viewer decides to record a first programme and a second programme being broadcast at overlapping times. Alternatively, a viewer may wish to record a first programme and time shift a second programme being broadcast at the same time as the first. This latter case will now be described in the context of a preferred embodiment, with reference to Figure 15.

[0208] For the purposes of this discussion, it is presumed that the user wishes to watch (and timeshift) programme A being received in transport stream X, having duration 90 minutes and commencing at 21h00; and to record programme B being received in transport stream Y, having duration 45 minutes and commencing at 21h30.

[0209] Having indicated earlier in a known manner that programme B is to be recorded, the user indicates at 21h00 that he desires to watch programme A, for example by selecting programme A from an electronic programme guide. A PLAYER/RECORDER logical demultiplexer device, DEMUX_PLAY_RECORD 6240 (as described above), is allocated to the demultiplexing of the service forming programme A; DEMUX_PLAY_RECORD determines, using the process described above, that the first physical demultiplexer, HW_DEMUX_0 6302, is available to demultiplexer from a tuner. A first tuner, TUNER_0 2016, is tuned to the frequency of transport stream X, and the source of HW_DEMUX_0 6302 is set to TUNER_0 2016 using the DEMUX_SET_SOURCE command. The PIDs of the el-

ementary streams belonging to programme A are obtained from the SI Engine 3540 (described above) and passed to HW_DEMUX_0 6302 which commences demultiplexing of those PIDs. The extracted elementary stream packets are descrambled (as described below) and presented to the user in a known manner.

[0210] At 21h30, a RECORDER logical demultiplexer device, DEMUX_RECORD 6220, is allocated to the demultiplexing of the service forming programme B. The DEMUX_RECORD 6220 determines, as described above, that HW_DEMUX_0 6302 is performing a demultiplexing operation on the transport stream being received in transport stream X, but that a second physical demultiplexer, HW_DEMUX_1 6304, is available. A second tuner, TUNER_1 2016, is tuned to the frequency of transport stream Y, and the source of HW_DEMUX_1 6304is set to TUNER_1 2016.

[0211] The PIDs belonging to programme B are determined as above and passed to HW_DEMUX_1 6304 which commences demultiplexing of those PIDs. The extracted elementary streams are then remultiplexed and stored on the hard disk in a known manner.

[0212] For the purposes of this discussion we now suppose that, at 21h45, the user decides that he wishes to postpone viewing of the remaining 45 minutes of programme A, and indicates this by pressing a PAUSE button on a remote control. DEMUX_PLAY_RECORD 6240 ceases the descrambling of the extracted elementary streams, causes the SERVICE device 3736 allocated to it to freeze the video display on screen and commences the remultiplexing of the elementary streams using the TS_REMUX device 3766 allocated to it. The remultiplexed stream is stored on the hard disk in a known manner.

[0213] At 22h00, the user indicates that he wishes to view the remaining 45 minutes of programme A by pressing a resume button on a remote control. A PLAYER logical demultiplexer, DEMUX_PLAY 6220, is allocated to the demultiplexing of the stored portion of programme A; DEMUX_PLAY 6220 determines that HW_DEMUX_0 6302 and HW_DEMUX_1 6304 are not available to demultiplex a stream from the hard disk, and a third physical demultiplexer, HW_DEMUX_2 (not shown), is selected and its source set to the hard disk (via a FIFO) in a known manner. The stored portion of programme A is read, demultiplexed and descrambled from the hard disk by DEMUX_PLAY until 22h45, while DEMUX_PLAY_RECORD continues to demultiplex and remultiplex programme A from TUNER_0 2016 until 22h30. DEMUX_RECORD 6230 ceases demultiplexing and remultiplexing programme B from TUNER_1 2018 at 22h15.

[0214] If the user decides subsequently to view programme B, DEMUX_PLAY 6220 performs the demultiplexing operations, and so on, as described above.

Control word device

[0215] In known systems, the control words are extracted from the ECMs by the conditional access smartcard housed in the card reader and then continue to be managed by other items of hardware, that is to say at a low level. In embodiments of the invention, the control words continue to be extracted from the ECMs by the smartcard, but further processing of the conditional access data is handled at a higher level in the architecture.

[0216] Preferred embodiments comprise a further software device, the control word (CW) device 3760, for managing descrambling operations performed by the descrambler in the DDR 2010.

[0217] As mentioned above, the Service Information (SI) Engine 3540 loads and monitors common Digital Video Broadcasting (DVB) or Program System Information Protocol (PSIP) tables and puts them into a cache. Data contained in the tables includes Conditional Access Tables (CATs) from which the PIDs for Entitlement Control Messages (ECMs), each containing an encrypted control word and access criteria for a programme component, can be ascertained. To descramble incoming programme components, the relevant control words need to be loaded to the conditional access smartcard (CA smartcard) 2062 where they can be decrypted, using the key received in an Entitlement Management Message (EMM), and then used in the DDR unit 2010 for descrambling.

[0218] The handling of ECMs will now be further described with reference to Figure 16.

[0219] When a service is to be descrambled, a CA kernel 6402 in the middleware (for example, forming part of the virtual machine) receives an event from elsewhere in the system (for example, from the Zapping application 3146) identifying the channel upon which the service to be descrambled is being received. The CA kernel 6402 retrieves from the data cached by the SI engine 3540 the PIDs of the ECMs relevant to the service to be descrambled and instructs the MLOAD device 3738 in the DLI to isolate the ECMs 6420 as they are received in the programme data stream. The isolated ECMs are then routed by the CA kernel 6402 to the conditional access smartcard 2062 which operates under the control of an LCARD device 3740. The conditional access smartcard extracts control words from the ECMs in a known manner, if it has the rights to do so, and the extracted code words are passed by the CA kernel 6402 to the Control Word (CW) device 3760 in the DLI. As a result, the handling of the control words is not visible at a hardware level, resulting in increased security. This feature is of particular benefit when using a dedicated tuner for the reception of conditional access data, as will be described below, since it requires no changes to be made to the hardware level.

[0220] As indicated above, the role of the CW device is to manage descrambling operations performed by the physical descrambler(s). In the preferred embodiment,

the CW device is instantiated, as described above, and each instance of the device is a client of the (or one of the) descrambler(s). The CW device allocates a descrambler channel, identified by a descrambler channel identifier DESCR_ID, to each requested descrambling operation for which different access conditions apply (that is, requiring the use of different control words).

[0221] For a particular descrambling operation, the CW device is responsible for the following tasks:

- receiving a request indicating that data is to be descrambled;
- allocating a descrambler channel to the descrambling operation;
- selecting the type of descrambling operation to be performed (for example, DVB CS, DES and Triple DES);
- passing the control keys to the descrambler;
- flushing control keys from a descrambler channel, when necessary; and
- closing a channel when it is no longer required.

[0222] In addition, the CW device determines the maximum number of descrambler channels which can be allocated at any time, based on the capabilities of the descramblers, and it notifies the CA kernel if a request for a descrambling operation cannot be accommodated.

Multiple tuners

[0223] The provision of two or more tuners in preferred embodiments will be described below after a brief review of provisions for conditional access in prior art systems.

[0224] In known systems, an MPEG transport stream carries a limited number of audio and video programmes, the number being dependent on factors such as the degree of data compression, and so on. Administrative data, including conditional access data, for all of the programmes being broadcast from a head-end is inserted into each of the transport streams, so that a receiver/decoder is able to determine for which programmes the rights to view are held, without tuning to each transport stream in turn. This provision of such data in each transport stream reduces the bandwidth available for content itself, and thus increases the number of transport streams, transponders, and so on, necessary for the distribution of a bouquet.

[0225] In a preferred embodiment, a head-end is provided which is capable of broadcasting this administrative data (including conditional access data) in a single transport stream only. It is indicated above that a preferred embodiment comprises two tuners. In a preferred embodiment, one of these tuners (or, in a particularly preferred embodiment, an additional tuner) is dedicated to the task of receiving this administrative data.

[0226] In the description above referring to Figures 1, 2 and 3, a single multiplexer 1030 is described for constructing a transport stream for broadcast to users over a linkage 1022. The multiplexer 1030 in fact represents a plurality of multiplexers which construct the plurality of transport streams necessary for broadcasting a bouquet. The provision of an additional multiplexer for broadcasting administrative data in a preferred embodiment is now further described with reference to Figure 17.

[0227] In addition to the multiplexers 1030, an administrative data multiplexer 1030' is provided which receives the encrypted EMMs from the SAS 5200, encrypted ECMs from the second encrypting unit 5102, datastream description tables (for example, the PATs and the PMTs), update packets for the CAT table and any other data of a similar type, which it uses to assemble an administrative data transport stream. This transport stream has a broadcast route 600' (which may be the same as the broadcast routes 600). In this embodiment, the administrative data is not inserted into the transport streams constructed by the multiplexers 1030.

[0228] When a receiver/decoder is installed at a user's premises, the frequency of the administrative data transport stream is programmed during initialisation. In an alternative embodiment, the receiver/decoder is adapted to scan frequencies for a transport stream identifying itself as the administrative data transport stream.

[0229] In further embodiments, the conditional access data is not broadcast (and therefore received) on a dedicated channel, but on a channel which also carries some programme data. In yet further embodiments, channel-hopping is implemented, such that channel upon which the conditional access data is broadcast changes.

[0230] In a preferred embodiment of a receiver/decoder intended to be used to receive transport streams broadcast by a head-end as described above, there is provided a first tuner 2016 tunable to the administrative data transport stream and a second tuner 2018 (or, in a particularly preferred embodiment, second and third tuners) tunable to transport streams comprising programme data.

[0231] It will be readily understood that, in order to demultiplex both programme data and administrative data simultaneously, the preferred embodiment of receiver/decoder must be provided with at least two physical demultiplexers in the DDR unit 2010, a first to filter programme stream PIDs from one transport stream and a second to filter the related administrative data PIDs from the administrative data transport stream. In fact, as indicated above, a particularly preferred embodiment comprises three physical demultiplexers.

[0232] The precise details of the implementation of the various functions described above, and their distribution between hardware and software, are a matter of choice for the implementer and will not be described in detail. It is, however, noted that dedicated integrated circuits capable of performing the operations required in the receiver/decoder are commercially available or can

be readily designed, and these can be used as the basis for a hardware accelerator, or more preferably modified to produce a dedicated hardware accelerator, to implement various of the operations required, thereby reducing the processing power required to run the software. However, the operations required may be implemented in software if sufficient processing power is available.

[0233] The modules and other components have been described in terms of the features and functions provided by each component, together with optional and preferable features. With the information given and specifications provided, actual implementation of these features and the precise details are left to the implementer. As an example, certain modules could be implemented in software, preferably written in the C programming language and preferably compiled to run on the processor used to run the application; however, some components may be run on a separate processor, and some or all components may be implemented by dedicated hardware.

[0234] The above modules and components are merely illustrative, and the invention may be implemented in a variety of ways, and, in particular, some components may be combined with others which perform similar functions, or some may be omitted in simplified implementations. Hardware and software implementations of each of the functions may be freely mixed, both between components and within a single component.

[0235] It will be readily understood that the functions performed by the hardware, the computer software, and such like are performed on or using electrical and like signals. Software implementations may be stored in ROM, or may be patched in FLASH.

[0236] It will be understood that the present invention has been described above purely by way of example, and modification of detail can be made within the scope of the invention.

[0237] Each feature disclosed in the description, and (where appropriate) the claims and drawings may be provided independently or in any appropriate combination.

Claims

1. A logical device for a receiver/decoder, preferably excluding a software device which corresponds to a single hardware device.

2. A logical device according to Claim 1, comprising means for binding the logical device to a further device.

3. A logical device according to Claim 2, wherein the binding means is adapted to bind the logical device to a plurality of further devices.

4. A logical device according to Claim 2 or Claim 3, wherein the binding means is adapted to bind the logical device to a software device.

5. A logical device according to any of the preceding claims, adapted to manage a demultiplexing operation.

6. A logical device according to Claim 5, comprising means for restricting functionality offered by the device.

7. Apparatus for a receiver/decoder, the apparatus being provided with at least one function description describing a function supported by one or more processes in use of the receiver/decoder, and means for loading a value with respect to the or each function description for use in said one or more processes.

8. Apparatus according to Claim 7 wherein each function description comprises a set of one or more data fields which together characterise the described function in use of the receiver/decoder.

9. Apparatus according to Claim 8 wherein the availability of one or more of said processes is determined by a value in at least one data field associated with that process.

10. Apparatus according to Claim 9 wherein said availability is determined by the presence or absence of said value in the at least one data field.

11. Apparatus according to any of Claims 7 to 10, wherein each process, and thus each function, is supported in use by one or more devices, the apparatus being provided with:

    i) at least one device description in respect of one or more of said devices;
    ii) means for loading device-specific data with respect to said device description;
    iii) means to detect the presence of at least one device in the receiver/decoder which supports a function described by the function description; and
    iii) means to load a function identifier for that function description as part of a device identifier for use in communicating with the at least one device in use of the receiver/decoder.

12. Apparatus according to any of Claims 7 to 11, wherein the function supported by one or more processes comprises a demultiplexing function.

13. Apparatus according to Claim 12 wherein said means for loading a value is adapted to load multiple values and said multiple values comprise a set

of identifiers for input sources of multiplexed signals.

14. Apparatus according to any of Claims 11, 12 or 13 wherein said means to detect the presence of at least one device comprises means to detect the presence of each of a plurality of devices in the receiver/decoder which together support the function described by the function description and said means to load the function identifier for that function description as part of a device identifier is adapted to load the same function identifier as part of the device identifier in respect of each of said plurality of devices.

15. Apparatus according to any of Claims 11 to 14 wherein the apparatus is adapted to modify a function description in accordance with devices detected.

16. Apparatus according to any of Claims 11 to 15 wherein the apparatus is provided with at least two different function descriptions and the means to load a function identifier is adapted to load a function identifier for each of at least two different function descriptions to provide at least two different device identifiers for use in communicating with a common device in use of the receiver/decoder.

17. Apparatus according to any of Claims 7 to 16 which apparatus comprises control signal management means for managing signals for controlling one demultiplexing device to demultiplex at least first and second data streams over a common time period.

18. Apparatus according to any of Claims 7 to 17 which apparatus comprises control signal management means for managing signals for controlling one or more remultiplexing devices to remultiplex at least first and second data streams for recording over a common time period.

19. A computer program product comprising apparatus according to any of Claims 7 to 18.

20. A receiver/decoder comprising apparatus according to any of Claims 7 to 18.

21. A method of operating a receiver/decoder, which method comprises creating at least one function description describing a function supported by one or more processes in use of the receiver/decoder, and loading a value with respect to the or each function description for use in said one or more processes.

22. A method according to Claim 21 wherein said loading is done at first initialisation of the receiver/decoder.

23. A method of using a receiver/decoder which comprises communicating with at least one device in providing a function, wherein a device identifier is used in communicating with the at least one device, which identifier comprises a part associated with the function and a part associated with the device.

24. Apparatus comprising means for instantiating a device for a receiver/decoder.

25. Apparatus according to Claim 24, wherein the instantiating means comprises means for providing generic information and means for providing instance-specific information to a device being instantiated.

26. Apparatus according to Claim 24 or Claim 25, wherein the instantiating means is adapted to instantiate a device statically.

27. Apparatus according to any of the claims 24 to 26, wherein the instantiated device is adapted to provide functionality to a client.

28. Apparatus according to any of the claims 24 to 27, wherein the instantiated device is adapted to provide functionality to a plurality of clients.

29. Apparatus according to any of the claims 24 to 27, wherein the instantiated device is capable of asynchronous communication.

30. Apparatus according to any of the claims 24 to 29, comprising means for receiving information relating to a new type of device to be instantiated.

31. Apparatus according to any of the claims 24 to 30, comprising means for determining a hardware environment.

32. Apparatus according to any of the preceding claims, wherein the instantiated device is a logical device as claimed in any of claims 1 to 6.

33. Apparatus for a receiver/decoder, the apparatus being provided with at least one device description and means for loading device-specific data with respect to said device description.

34. Apparatus according to Claim 33 wherein said means for loading device-specific data is adapted to load at least one value with respect to one or more device descriptions, said value comprising at least part of a device identifier for use in communicating with a device in use of the receiver/decoder.

35. Apparatus according to Claim 33 or 34, further comprising means for loading device-specific data with

respect to more than one respective copy of at least one of the device descriptions.

36. Apparatus according to Claim 35 wherein a value loaded to each device instance as device-specific data comprises at least part of a device identifier and is different from any value loaded as at least part of a device identifier to another copy of the same device description.

37. Apparatus according to any of Claims 34 to 36 wherein, in use of the receiver/decoder to provide a function, more than one different device is used to provide said function, wherein each device description with at least one value loaded provides a device instance, and wherein said at least part of a device identifier is common to at least one device instance for each of at least two different devices used to provide said function.

38. Apparatus according to any of the claims 33 to 37, wherein a value which can alternatively or additionally be loaded to one or more device descriptions as device-specific data comprises at least one value for a parameter of a procedure supported by a device to which the description relates.

39. Apparatus according to Claim 38 wherein said at least one value for a parameter of a procedure comprises an identifier for an authorised input to the device to which the description relates.

40. Apparatus according to Claim 39 wherein the device to which the description relates comprises a demultiplexing device.

41. Apparatus according to any of Claims 35 to 40, further comprising limiting means for limiting the number of copies of one or more device descriptions to a preset maximum.

42. Apparatus according to Claims 33 to 41, the apparatus being provided with at least one function description describing a function supported by one or more processes in use of the receiver/decoder, and means for loading a value with respect to the or each function description for use in said one or more processes.

43. A method of using a receiver/decoder to provide a function, which method comprises selecting an identifier for a device from at least two different identifiers available for that device and using the selected identifier in relation to communications with the device to provide the function.

44. A method of using a receiver/decoder to provide a function, which method comprises communicating with at least two different devices used in providing the function, using a different respective identifier in relation to communication with each of said two different devices, wherein said different respective identifiers share a common portion.

45. Apparatus for processing data, comprising means for operating a demultiplexer to demultiplex a plurality of services simultaneously.

46. Apparatus according to Claim 1, wherein the demultiplexer operating means comprises means for allocating a respective logical demultiplexer as claimed in any of claims 1 to 6 to each service to be demultiplexed.

47. Apparatus for controlling a demultiplexing process in a receiver/decoder, which apparatus comprises control signal management means for managing signals for controlling one demultiplexing device to demultiplex at least first and second data streams over a common time period.

48. Apparatus according to Claim 47, wherein said control signal management means is adapted to maintain a first family of devices for use together in controlling the demultiplexing device to demultiplex said first data stream, and to maintain a second family of devices for use together in controlling the demultiplexing device to demultiplex said second data stream.

49. Apparatus according to Claim 48 wherein the devices of each family are each allocated an identifier which has at least a common portion for all the devices of a family, said common portion for the first family being different from said common portion for the second family, for use in co-ordinating processes performed by the devices of each family in controlling the demultiplexing device to demultiplex a respective data stream.

50. Apparatus according to any of Claims 47 to 49, further comprising at least one remultiplexing device for remultiplexing each of said at least two data streams for recording.

51. Apparatus for a receiver/decoder, comprising apparatus according to any of Claims 47 to 50, which apparatus for a receiver/decoder further comprises at least two inputs for connection to respective channels and correlation means for correlating a signal received at a first of said inputs with a signal received at a second of said inputs.

52. A method of controlling a demultiplexing process in a receiver/decoder, which method comprises sending one or more control signals to one demultiplex-

ing device, which control signal(s) identify at least first and second data streams to be demultiplexed over a common time period.

53. A method according to Claim 52 which further comprises maintaining a first family of devices for use together in controlling the demultiplexing device to demultiplex said first data stream, and maintaining a second family of devices for use together in controlling the demultiplexing device to demultiplex said second data stream.

54. A method according to Claim 52 which further comprises allocating an identifier to each device of each family, which identifier has at least a common portion for all the devices of a family, said common portion for the first family being different from said common portion for the second family, for use in co-ordinating processes performed by the devices of each family in controlling the demultiplexing device to demultiplex a respective data stream.

55. A control word device for managing a descrambling operation, the device being implemented in software.

56. Apparatus for a receiver/decoder comprising an interface for use in controlling descrambling equipment to descramble signals.

57. Apparatus according to Claim 56 further comprising means to assign an identifier to an instance of use of the descrambling equipment to descramble a signal, which identifier is used in controlling the descrambling equipment in relation to that instance of use by means of the interface.

58. Apparatus according to Claim 57 wherein the means to assign an identifier comprises means to assign more than one identifier to the descrambling equipment over the same time period, each identifier being for use in controlling the descrambling equipment with respect to a respective instance of use of the descrambling equipment.

59. Apparatus according to Claim 58 wherein at least first and second instances of use of the descrambling equipment are subject to at least one different respective access condition for descrambling signals.

60. Apparatus according to any of Claims 57 to 59 wherein the apparatus is arranged to control supply of a decryption key to the descrambling equipment for use in a respective descrambling process.

61. Apparatus according to Claim 60 wherein the apparatus is arranged to coordinate supply of a decryption key, from means for receiving delivery of decryption keys, to the descrambling equipment, said co-ordination being done by use of the identifier assigned to each respective instance of use of the descrambling equipment.

62. Apparatus according to Claim 61 wherein the decryption key is received in a multiplexed information signal.

63. Apparatus according to any of Claims 56 to 62 wherein the interface is accessible to at least one device in a family of devices supporting a demultiplexing function.

64. Apparatus according to Claims 57 and 63 further comprising means to provide more than one instance of the same device in the family of devices, each instance so provided being related to at least one assigned identifier for an instance of use of the descrambling equipment.

65. Apparatus according to any of Claims 56 to 64, further comprising recording means for recording two or more programme data streams over a common time period.

66. A method of descrambling scrambled signals which method comprises the use of an interface in the control of descrambling equipment.

67. A method according to Claim 66 further comprising the steps of:

i) assigning an identifier to an instance of use of the descrambling equipment; and
ii) using the identifier in controlling the descrambling equipment in relation to said instance of use by means of the interface.

68. A method according to Claim 67 which comprises assigning a first identifier to a first instance of use of the descrambling equipment, assigning a second identifier to a second instance of use of the descrambling equipment, and using the first and second identifiers to distinguish the first and second instances of use in controlling the descrambling equipment, said first and second instances of use occurring over a common time period.

69. A method according to Claim 68 wherein said first and second instances of use of the descrambling equipment are subject to at least one different respective access condition for descrambling.

70. Apparatus for processing data, comprising means for recording a first service, means for simultaneously recording a second service and means for

playing back the first service and the second service at respective times chosen by a user.

71. Apparatus for a receiver/decoder comprising recording means for recording two or more programme data streams over a common time period.

72. A method of recording programme signals which method comprises recording two or more different programme data streams over a common time period.

73. Apparatus for processing data, comprising means for receiving service data on a first channel and means for receiving conditional access data relating to that service on a second channel.

74. Apparatus according to Claim 73, comprising means for causing the conditional access data receiving means to change the channel from which it receives the conditional access data.

75. A broadcast centre comprising service data broadcasting means adapted to broadcast service data excluding conditional access data on a first channel, and conditional access data broadcasting means for broadcasting conditional access data relating to the service on a second channel.

76. A broadcast centre according to Claim 75, comprising means for causing the conditional access data broadcasting means to change the channel upon which the conditional access data is broadcast.

77. A conditional access system comprising service data broadcasting means for broadcasting service data excluding conditional access data on a first channel, conditional access data broadcasting means for broadcasting conditional access data relating to the service on a second channel, service data receiving means for receiving the service data and conditional access data receiving means for receiving the conditional access data.

78. Apparatus for a receiver/decoder, for use in receiving and/or decoding signals received at the apparatus over more than one channel, wherein said apparatus comprises at least two inputs for connection to respective channels and correlation means for correlating a signal received at a first of said inputs with a signal received at a second of said inputs.

79. Apparatus according to Claim 78 further comprising detection means for detecting a channel identifier received in the signal at the first input, channel selection means for selecting a channel for connection to said second input, and control means to control the channel selection means to select a channel identified by the channel identifier for connection to said second input.

80. Apparatus according to either of Claims 78 or 79 wherein each respective channel has a different carrier frequency.

81. Apparatus according to any of Claims 78 to 80 wherein the signal received at the first input comprises at least primarily content data and the signal received at the second input comprises administrative data with respect to the content data.

82. Apparatus according to any of Claims 78 to 81, further comprising an interface for use in controlling descrambling equipment to descramble information signals.

83. A broadcast system for broadcasting related signals in multiplexed signal transport streams, which comprises broadcast means for broadcasting a first signal in a first multiplexed signal transport stream, broadcast means for broadcasting a second signal in a second multiplexed signal transport stream, and correlation signal transmission means for transmitting a correlation signal for use in correlating the related signals.

84. A broadcast system according to Claim 83 wherein the correlation signal comprises carrier frequency data for identifying a carrier frequency for at least one of the first and second transport streams.

85. A broadcast system according to either of Claims 83 or 84 wherein the correlation signal comprises at least one slot identifier for identifying a slot in one of the first and second transport streams.

FIG. 1

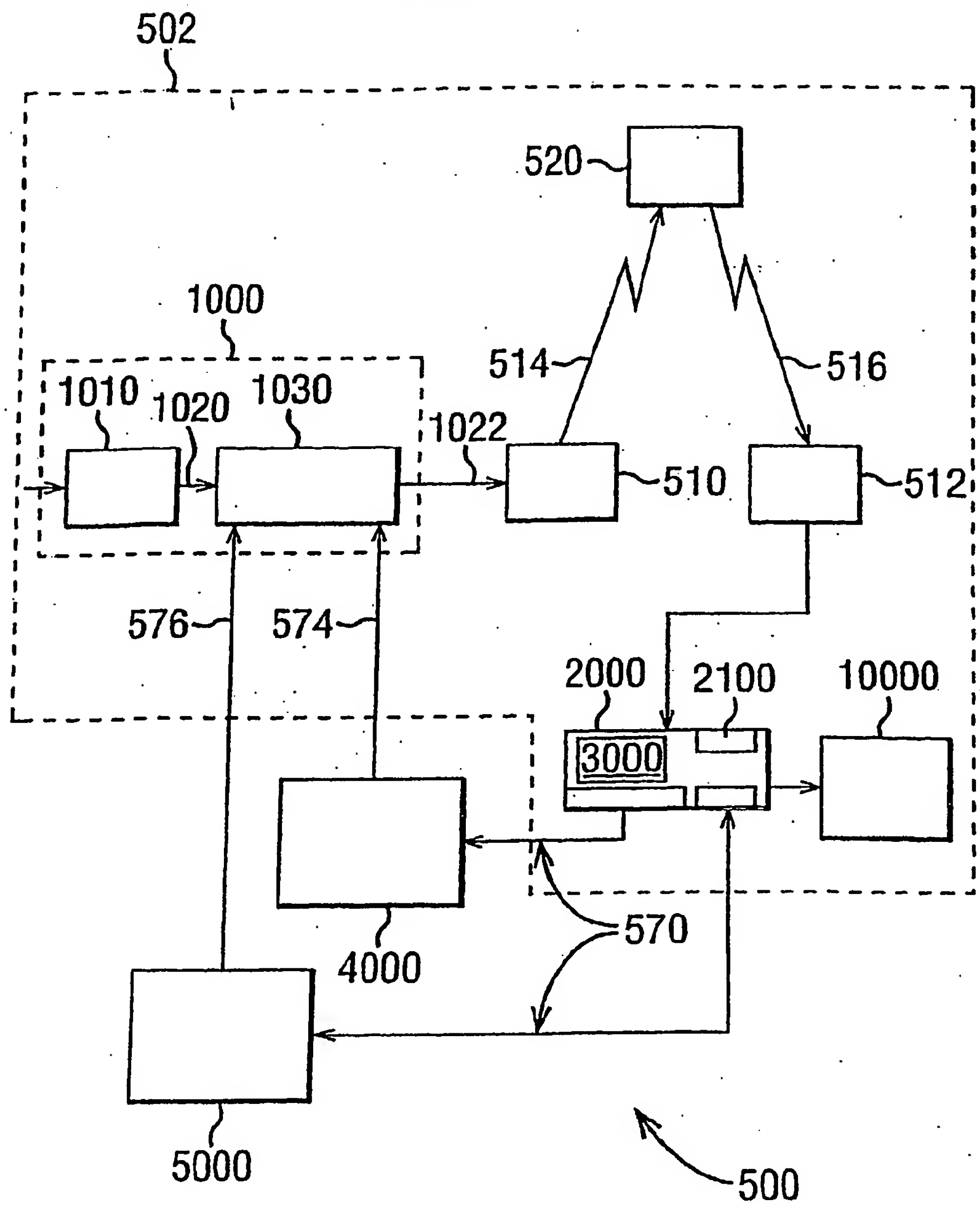

502
520
1000
514
516
1010
1020
1030
1022
510
512
576
574
2000
2100
10000
3000
570
4000
5000
500

FIG. 2

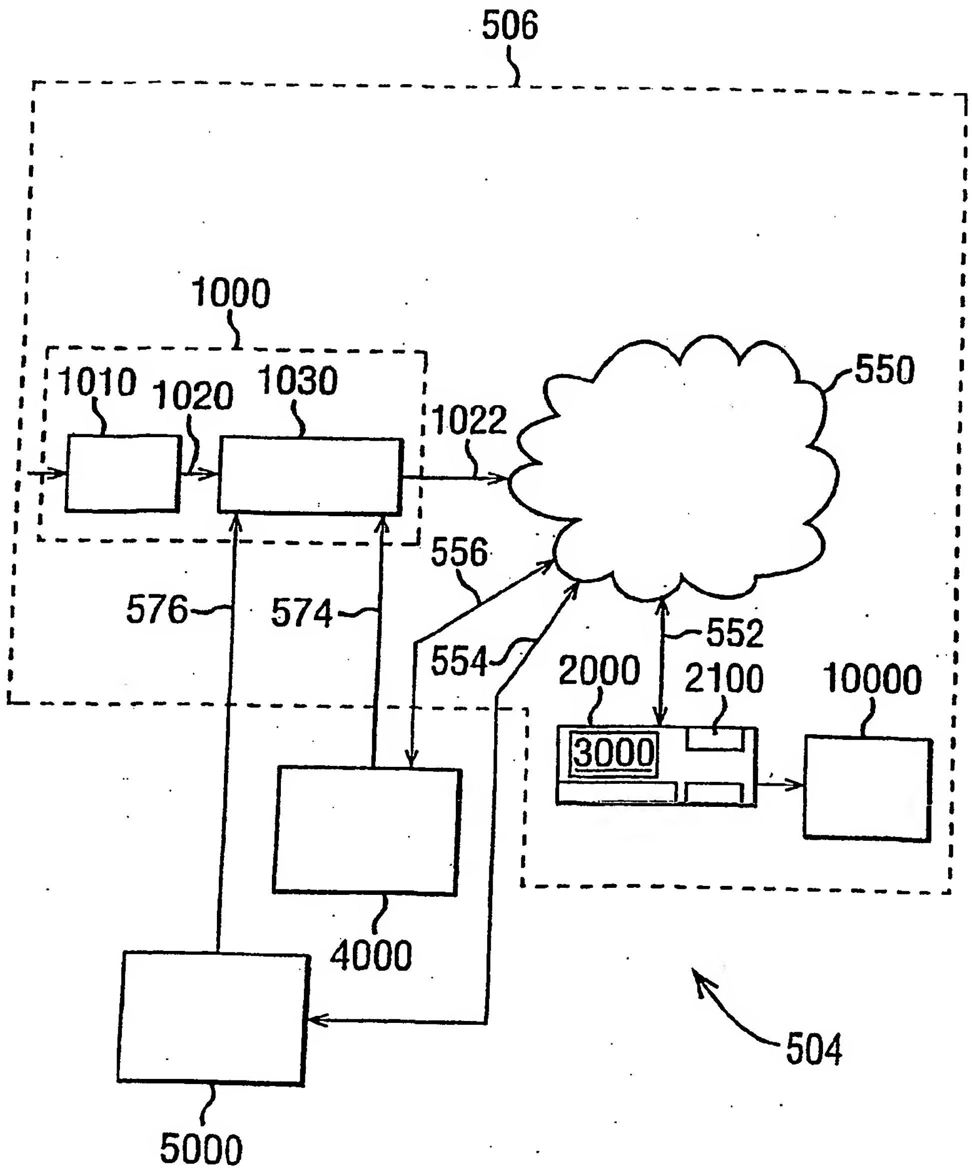
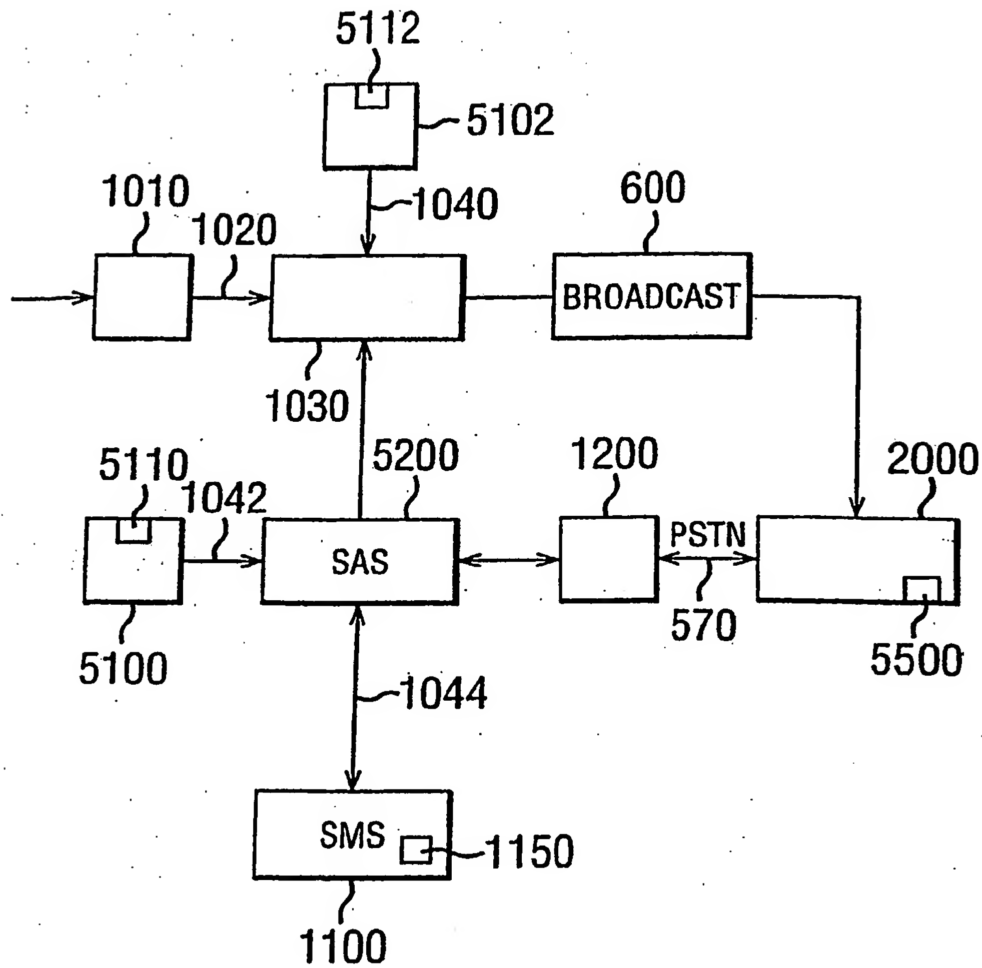

506
1000
1010
1020
1030
1022
550
556
576
574
552
554
2000
2100
10000
3000
4000
5000
504

FIG. 3


5112
5102
1010
1020
1040
600
BROADCAST
1030
5110
1042
5200
SAS
1200
PSTN
2000
570
5100
5500
1044
SMS
1150
1100

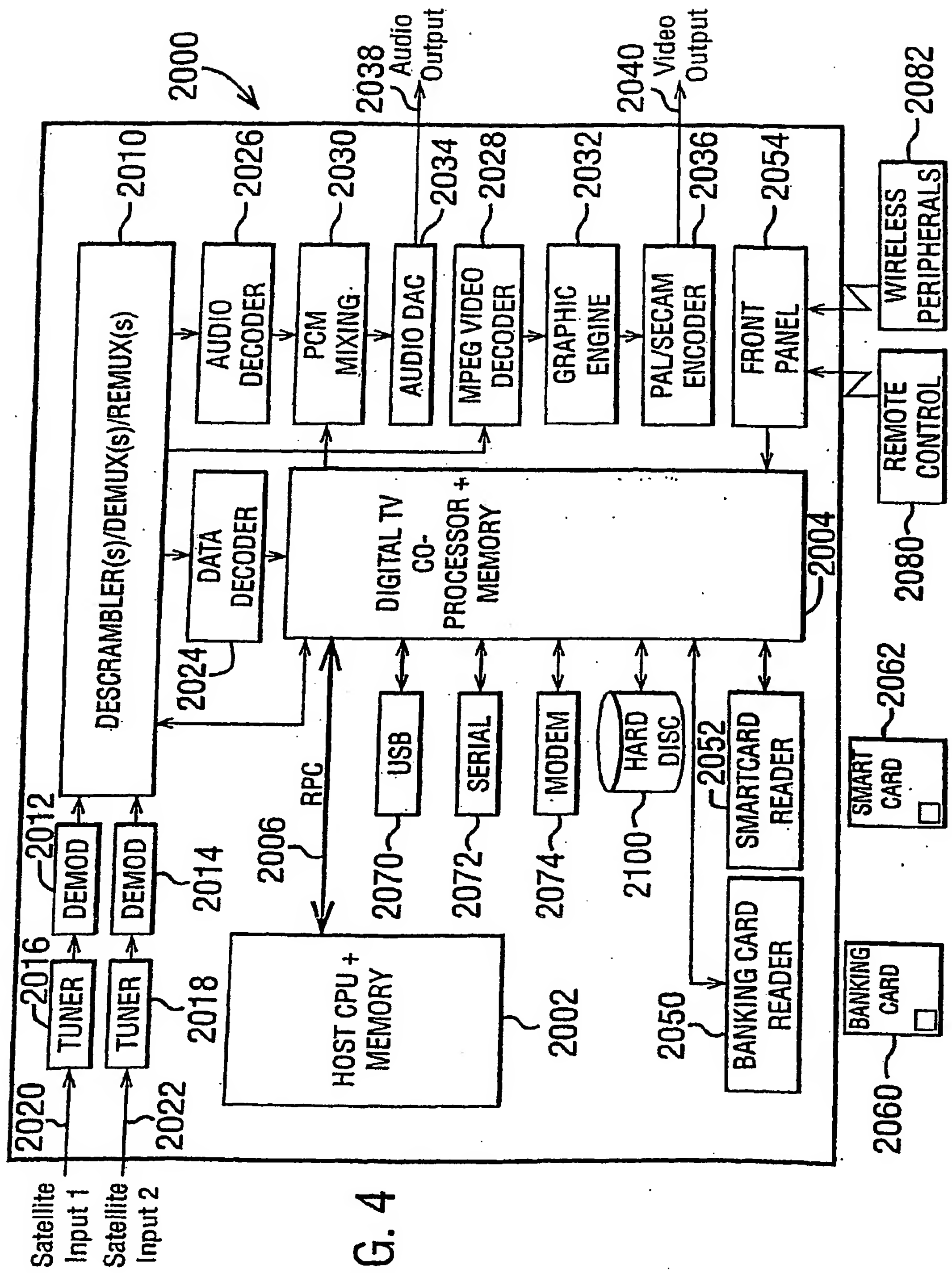
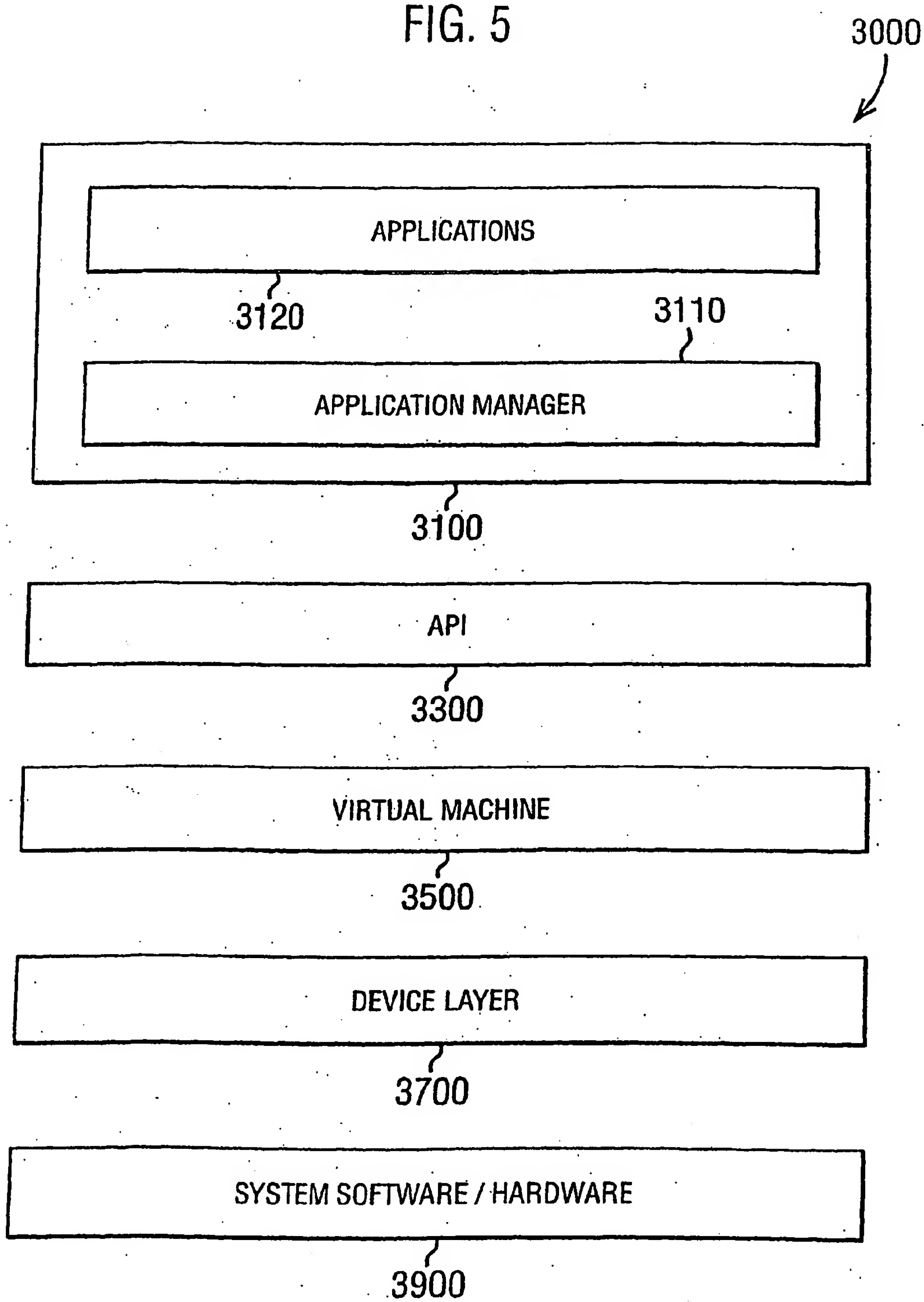

2000
DESCRAMBLER(s)/DEMUX(s)/REMUX(s)
2010
AUDIO DECODER
2026
PCM MIXING
2030
AUDIO DAC
2034
Audio Output
2038
MPEG VIDEO DECODER
2028
GRAPHIC ENGINE
2032
PAL/SECAM ENCODER
2036
Video Output
2040
FRONT PANEL
2054
WIRELESS PERIPHERALS
2082
REMOTE CONTROL
2080
DATA DECODER
2024
DIGITAL TV CO-PROCESSOR + MEMORY
2004
RPC
2006
USB
2070
SERIAL
2072
MODEM
2074
HARD DISC
2100
SMARTCARD READER
2052
SMART CARD
2062
BANKING CARD READER
2050
BANKING CARD
2060
HOST CPU + MEMORY
2002
DEMOD
2012
DEMOD
2014
TUNER
2016
TUNER
2018
2020
2022
Satellite Input 1
Satellite Input 2

FIG. 4

FIG. 5


3000
APPLICATIONS
3120
3110
APPLICATION MANAGER
3100
API
3300
VIRTUAL MACHINE
3500
DEVICE LAYER
3700
SYSTEM SOFTWARE / HARDWARE
3900

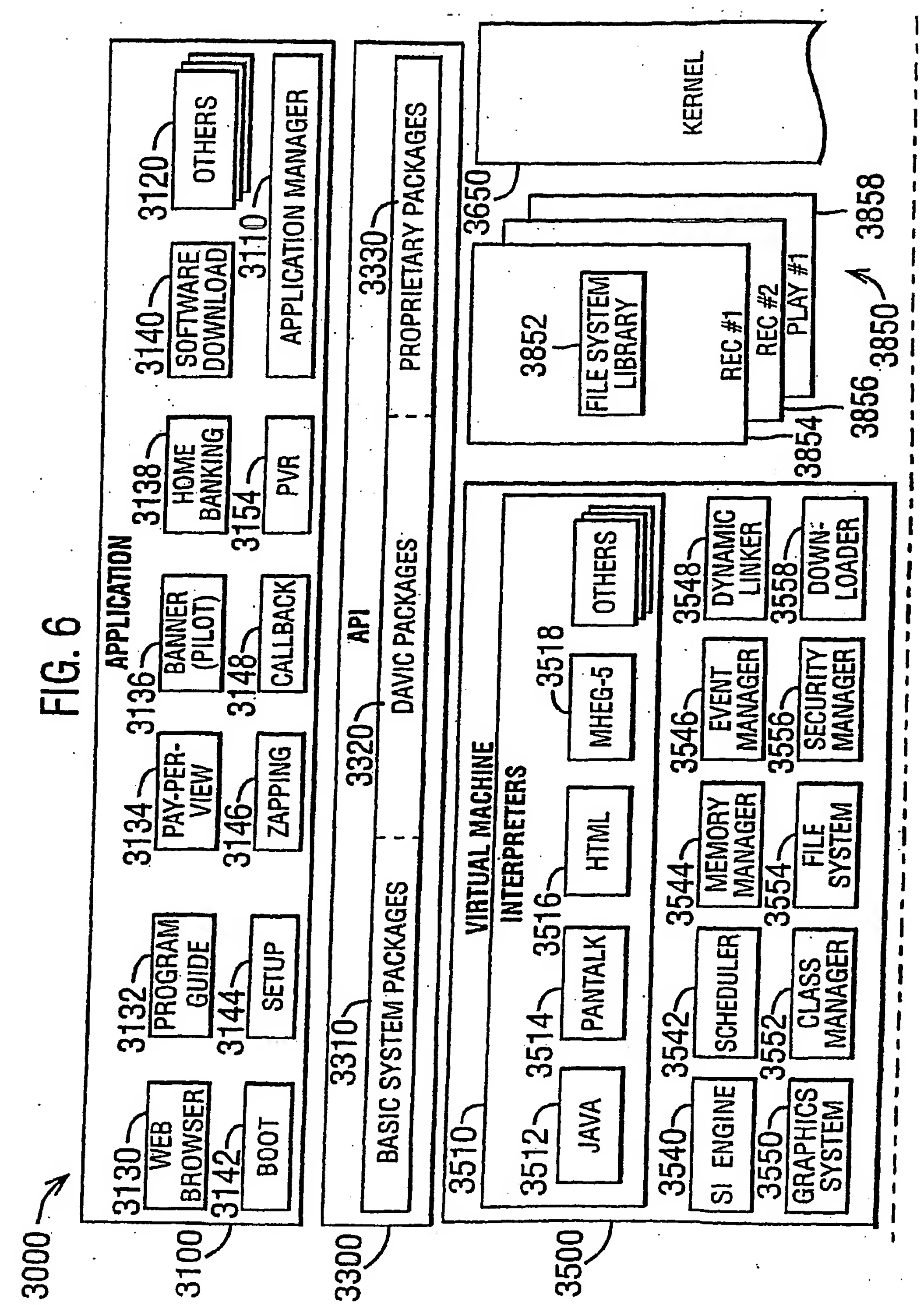

3000
3100
APPLICATION
3130 WEB BROWSER
3132 PROGRAM GUIDE
3134 PAY-PER-VIEW
3136 BANNER (PILOT)
3138 HOME BANKING
3140 SOFTWARE DOWNLOAD
3120 OTHERS
3142 BOOT
3144 SETUP
3146 ZAPPING
3148 CALLBACK
3154 PVR
3110 APPLICATION MANAGER
3300
API
3310 BASIC SYSTEM PACKAGES
3320 DAVIC PACKAGES
3330 PROPRIETARY PACKAGES
3500
3510 VIRTUAL MACHINE
INTERPRETERS
3512 JAVA
3514 PANTALK
3516 HTML
3518 MHEG-5
OTHERS
3540 SI ENGINE
3542 SCHEDULER
3544 MEMORY MANAGER
3546 EVENT MANAGER
3548 DYNAMIC LINKER
3550 GRAPHICS SYSTEM
3552 CLASS MANAGER
3554 FILE SYSTEM
3556 SECURITY MANAGER
3558 DOWN-LOADER
3650
KERNEL
3852 FILE SYSTEM LIBRARY
REC #1
REC #2
PLAY #1
3854
3856
3858
3850

FIG. 6

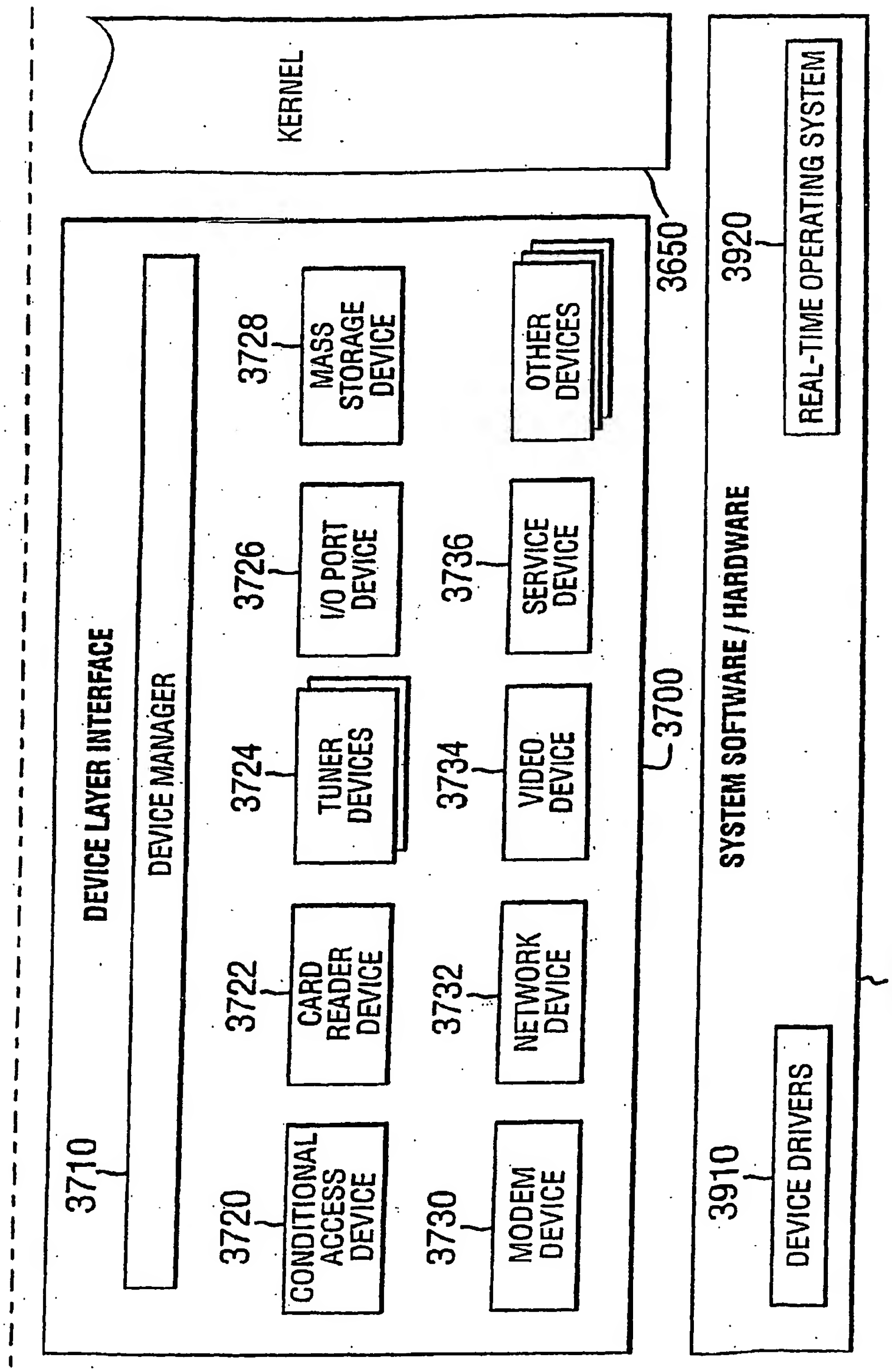

KERNEL
3650
DEVICE LAYER INTERFACE
DEVICE MANAGER
3710
3720
CONDITIONAL ACCESS DEVICE
3722
CARD READER DEVICE
3724
TUNER DEVICES
3726
I/O PORT DEVICE
3728
MASS STORAGE DEVICE
3730
MODEM DEVICE
3732
NETWORK DEVICE
3734
VIDEO DEVICE
3736
SERVICE DEVICE
OTHER DEVICES
3700
SYSTEM SOFTWARE / HARDWARE
3910
DEVICE DRIVERS
3920
REAL-TIME OPERATING SYSTEM
3900

FIG. 7

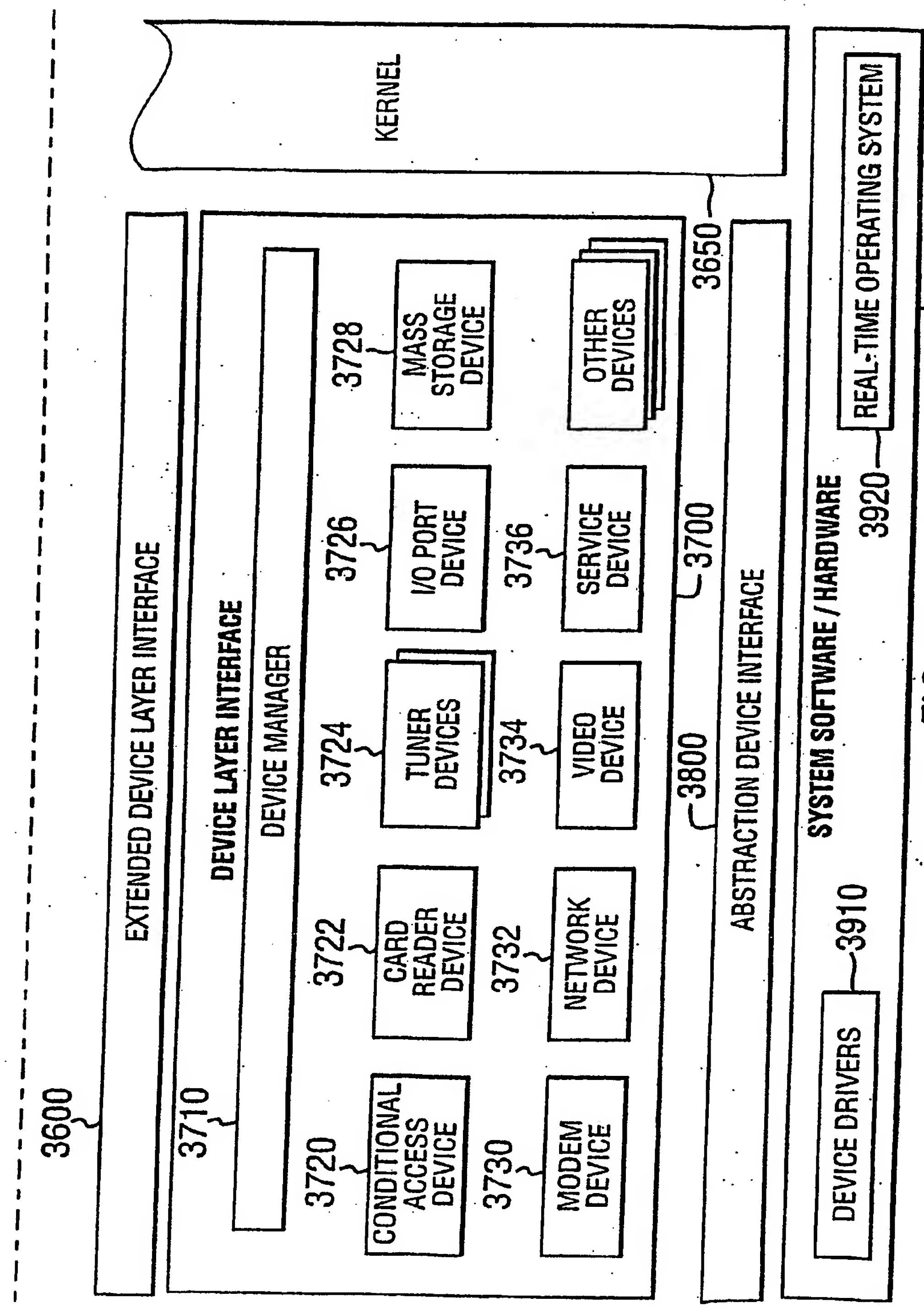

3600
EXTENDED DEVICE LAYER INTERFACE
KERNEL
3650
3710
DEVICE LAYER INTERFACE
DEVICE MANAGER
3720
CONDITIONAL ACCESS DEVICE
3722
CARD READER DEVICE
3724
TUNER DEVICES
3726
I/O PORT DEVICE
3728
MASS STORAGE DEVICE
3730
MODEM DEVICE
3732
NETWORK DEVICE
3734
VIDEO DEVICE
3736
SERVICE DEVICE
OTHER DEVICES
3700
3800
ABSTRACTION DEVICE INTERFACE
SYSTEM SOFTWARE / HARDWARE
DEVICE DRIVERS
3910
REAL-TIME OPERATING SYSTEM
3920
3900

FIG. 8

Fig. 9

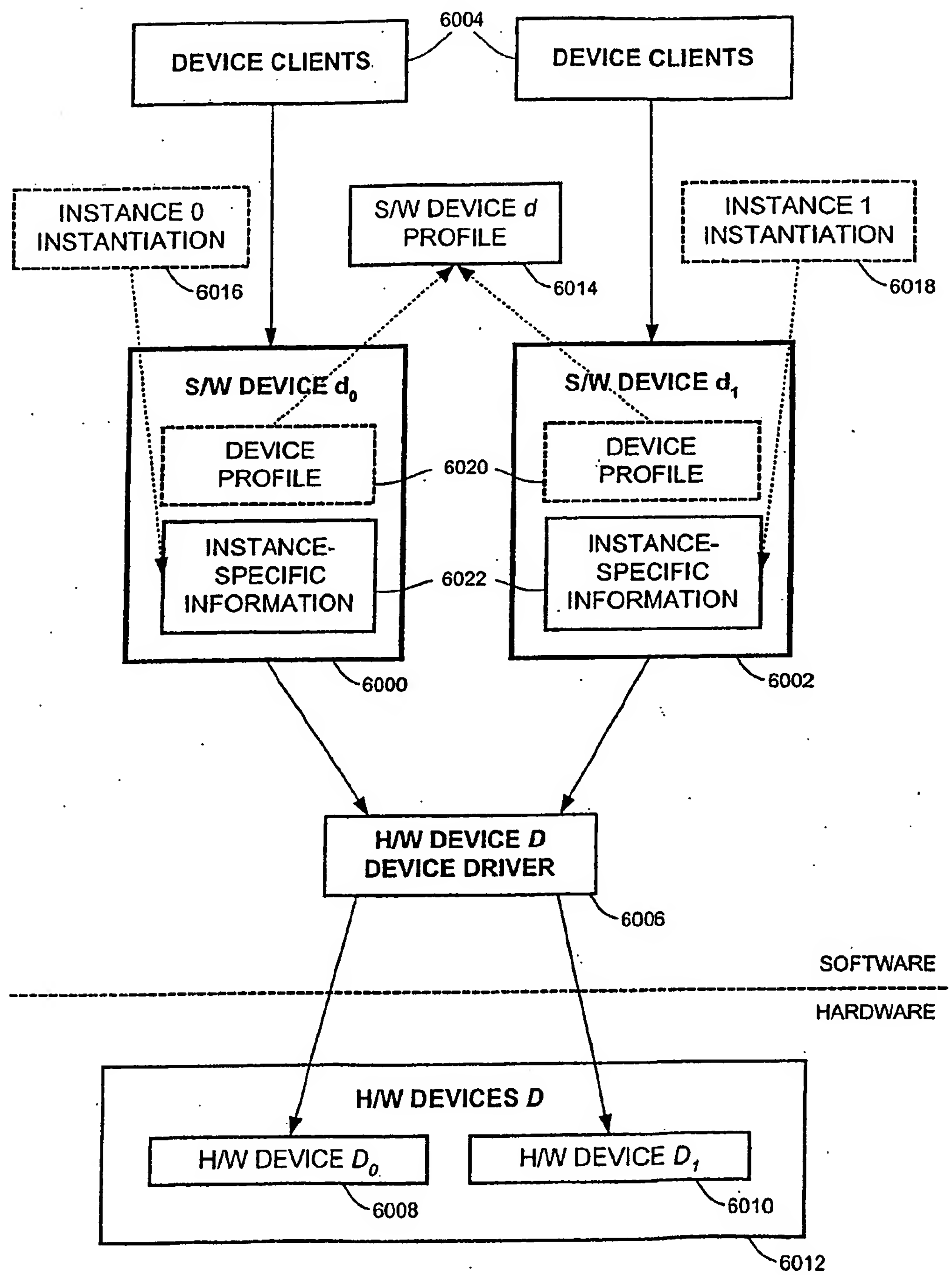

DEVICE CLIENTS
6004
DEVICE CLIENTS
INSTANCE 0 INSTANTIATION
6016
S/W DEVICE d PROFILE
6014
INSTANCE 1 INSTANTIATION
6018
S/W DEVICE d0
DEVICE PROFILE
INSTANCE-SPECIFIC INFORMATION
6020
6022
S/W DEVICE d1
DEVICE PROFILE
INSTANCE-SPECIFIC INFORMATION
6000
6002
H/W DEVICE D DEVICE DRIVER
6006
SOFTWARE
HARDWARE
H/W DEVICES D
H/W DEVICE D0
H/W DEVICE D1
6008
6010
6012

FIG. 10

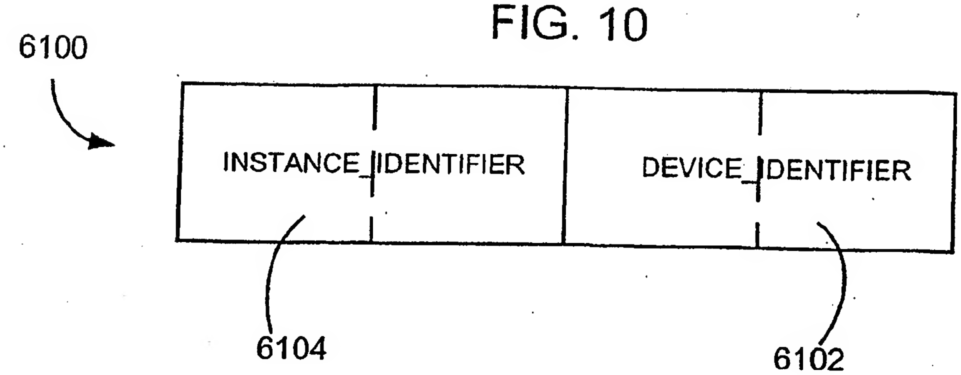

6100
INSTANCE_IDENTIFIER
DEVICE_IDENTIFIER
6104
6102

FIG. 11

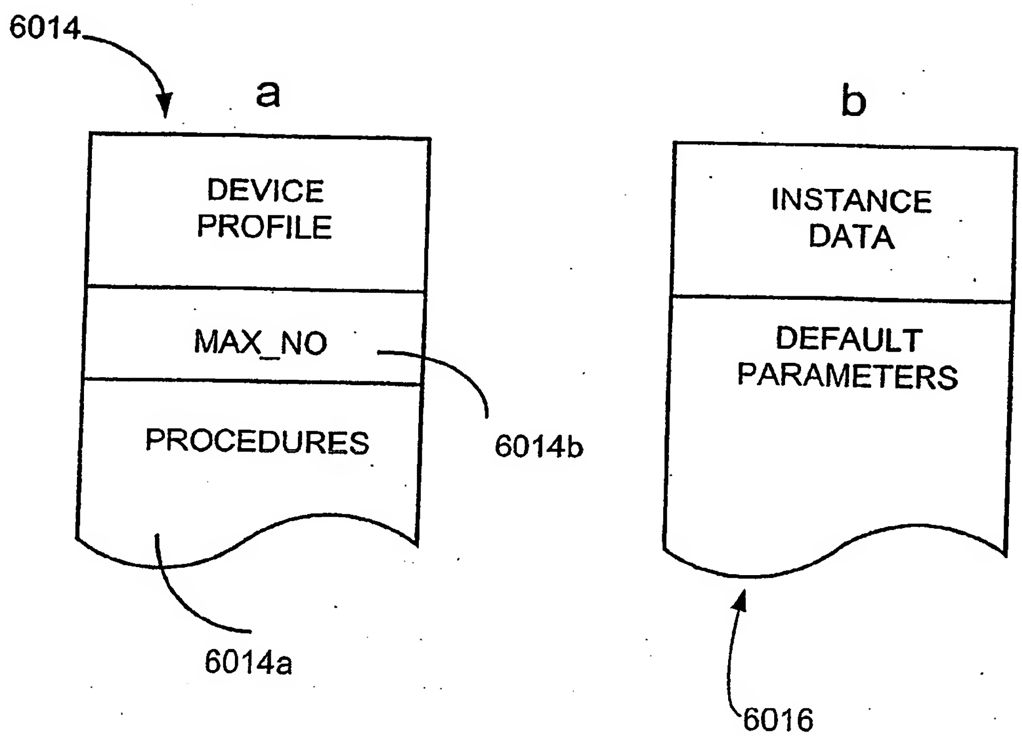

6014
a
DEVICE PROFILE
MAX_NO
PROCEDURES
6014b
6014a
b
INSTANCE DATA
DEFAULT PARAMETERS
6016

FIG. 12

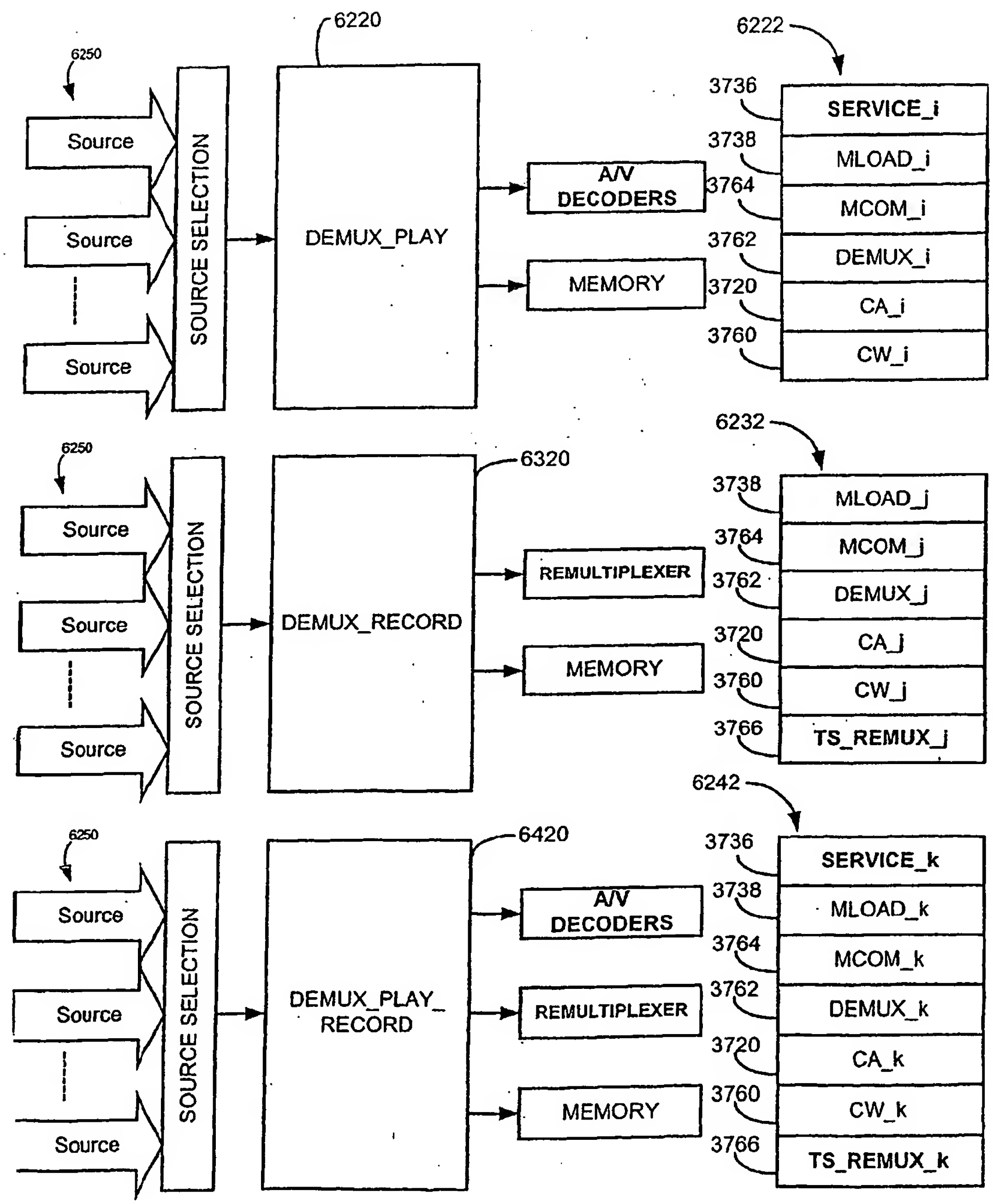
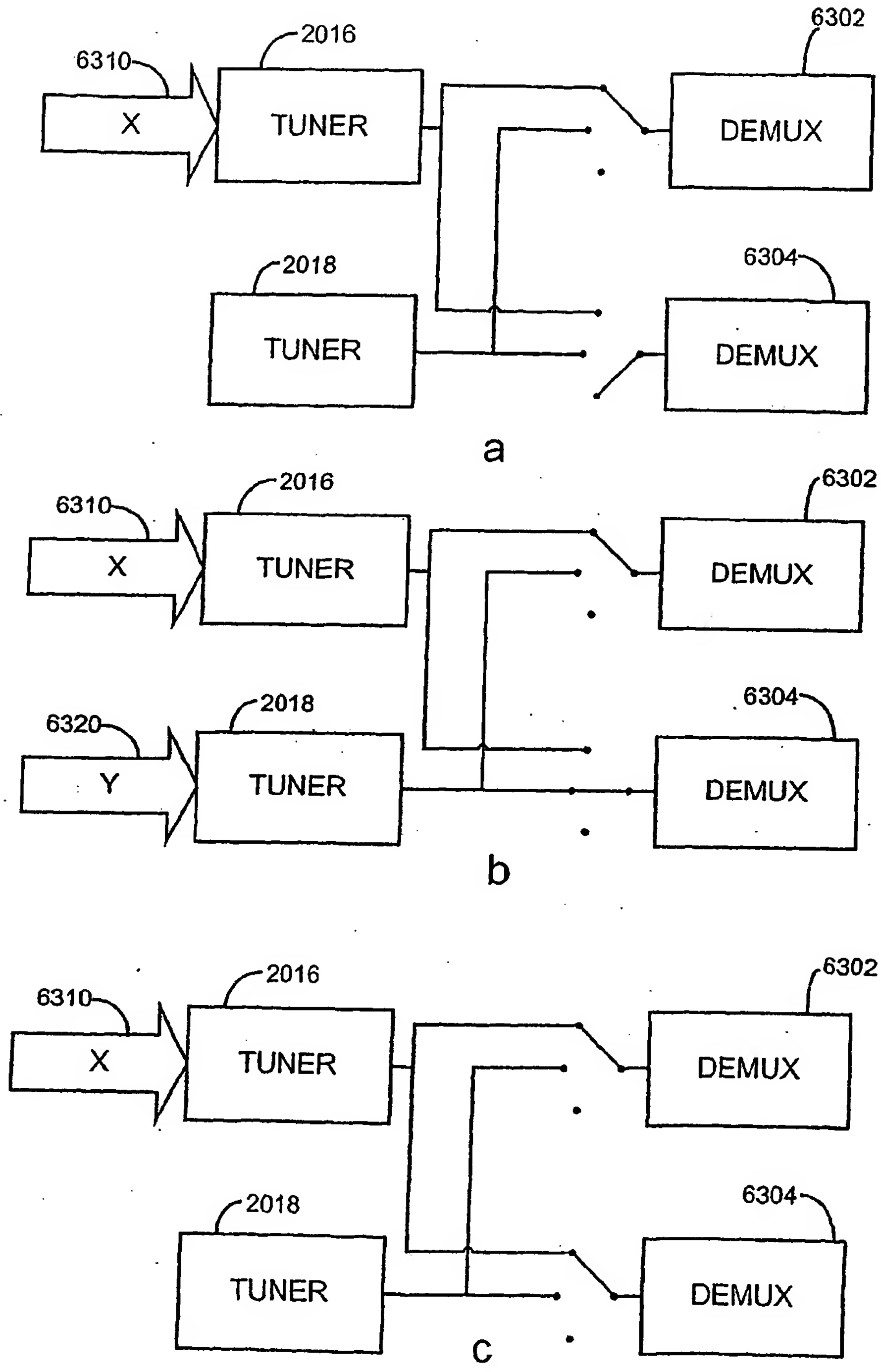

6220
6222
6250
Source
Source
Source
SOURCE SELECTION
DEMUX_PLAY
A/V DECODERS
MEMORY
3736
3738
3764
3762
3720
3760
SERVICE_i
MLOAD_i
MCOM_i
DEMUX_i
CA_i
CW_i
6232
6250
6320
Source
Source
Source
SOURCE SELECTION
DEMUX_RECORD
REMULTIPLEXER
MEMORY
3738
3764
3762
3720
3760
3766
MLOAD_j
MCOM_j
DEMUX_j
CA_j
CW_j
TS_REMUX_j
6242
6250
6420
Source
Source
Source
SOURCE SELECTION
DEMUX_PLAY_ RECORD
A/V DECODERS
REMULTIPLEXER
MEMORY
3736
3738
3764
3762
3720
3760
3766
SERVICE_k
MLOAD_k
MCOM_k
DEMUX_k
CA_k
CW_k
TS_REMUX_k

FIG. 13



6310
X
2016
TUNER
6302
DEMUX
2018
TUNER
6304
DEMUX
a
6310
X
2016
TUNER
6302
DEMUX
6320
Y
2018
TUNER
6304
DEMUX
b
6310
X
2016
TUNER
6302
DEMUX
2018
TUNER
6304
DEMUX
c

Figure 14

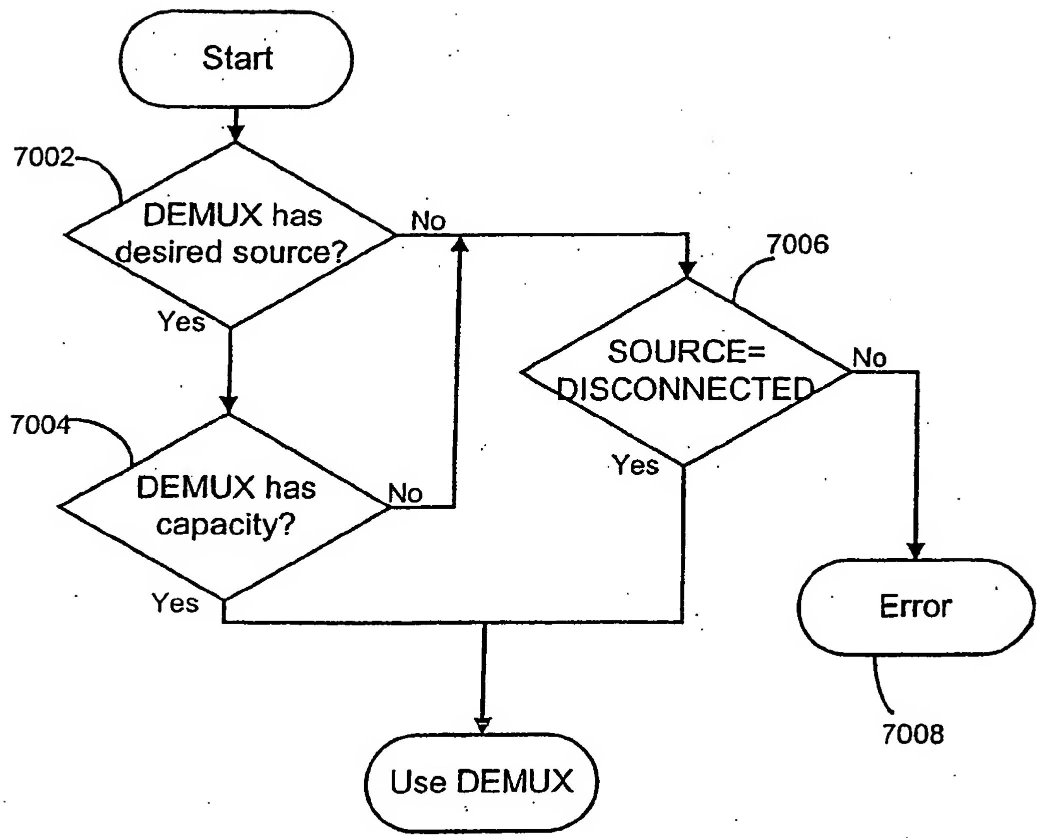

Start
7002
DEMUX has desired source?
No
Yes
7004
DEMUX has capacity?
No
Yes
7006
SOURCE= DISCONNECTED
No
Yes
Error
7008
Use DEMUX

FIG. 15

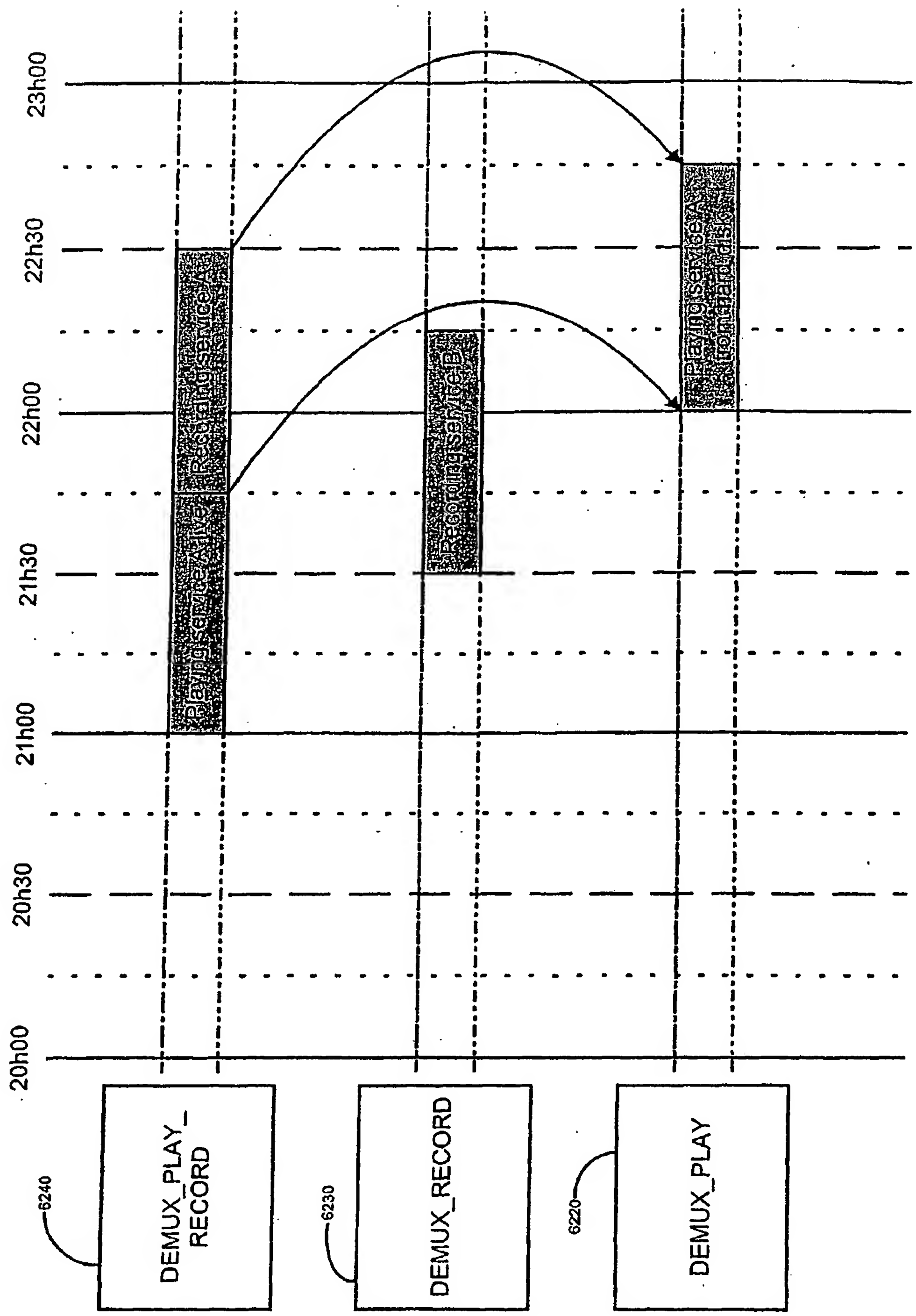

20h00
20h30
21h00
21h30
22h00
22h30
23h00
6240
DEMUX_PLAY_
RECORD
6230
DEMUX_RECORD
6220
DEMUX_PLAY

FIG. 16

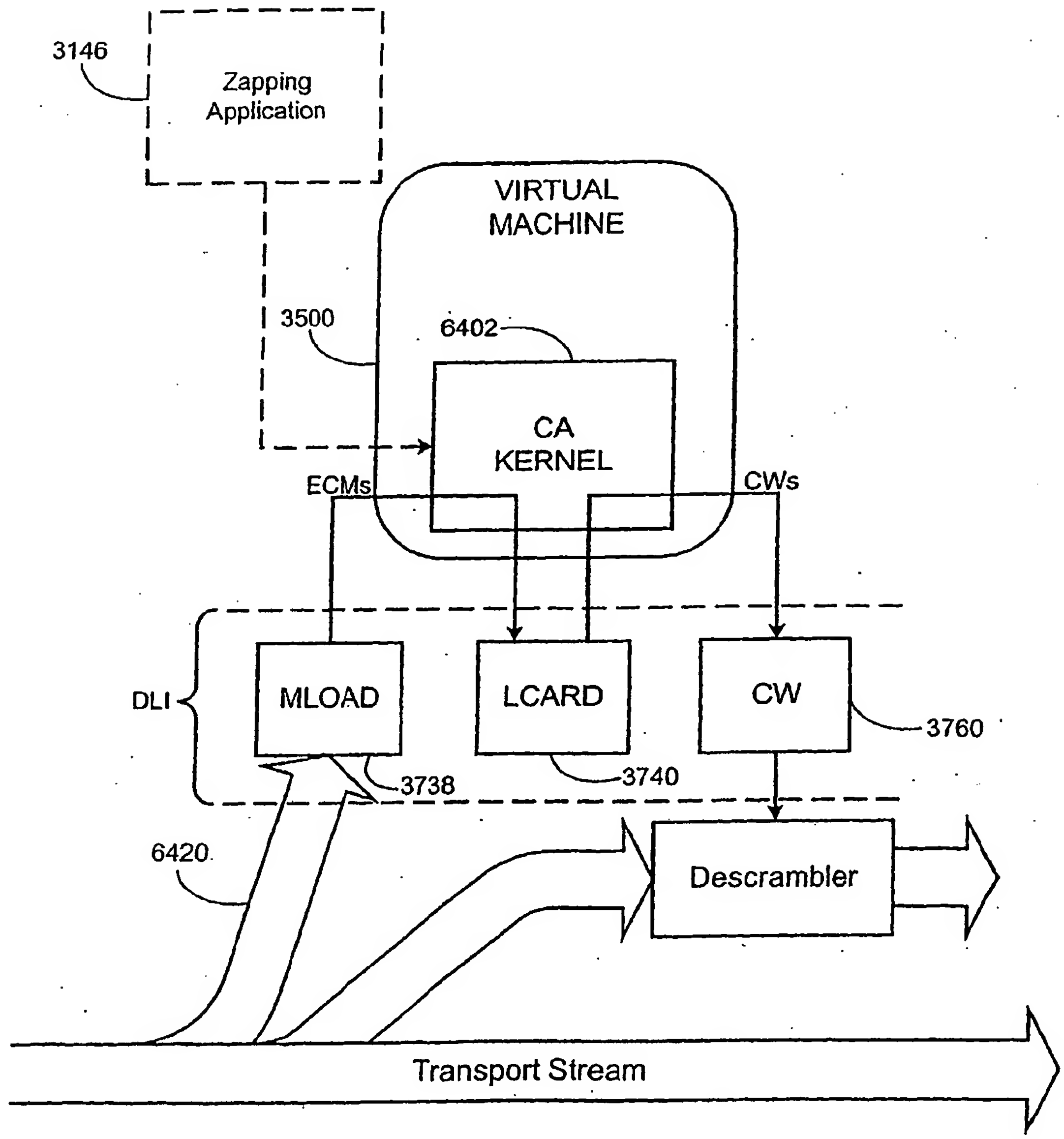
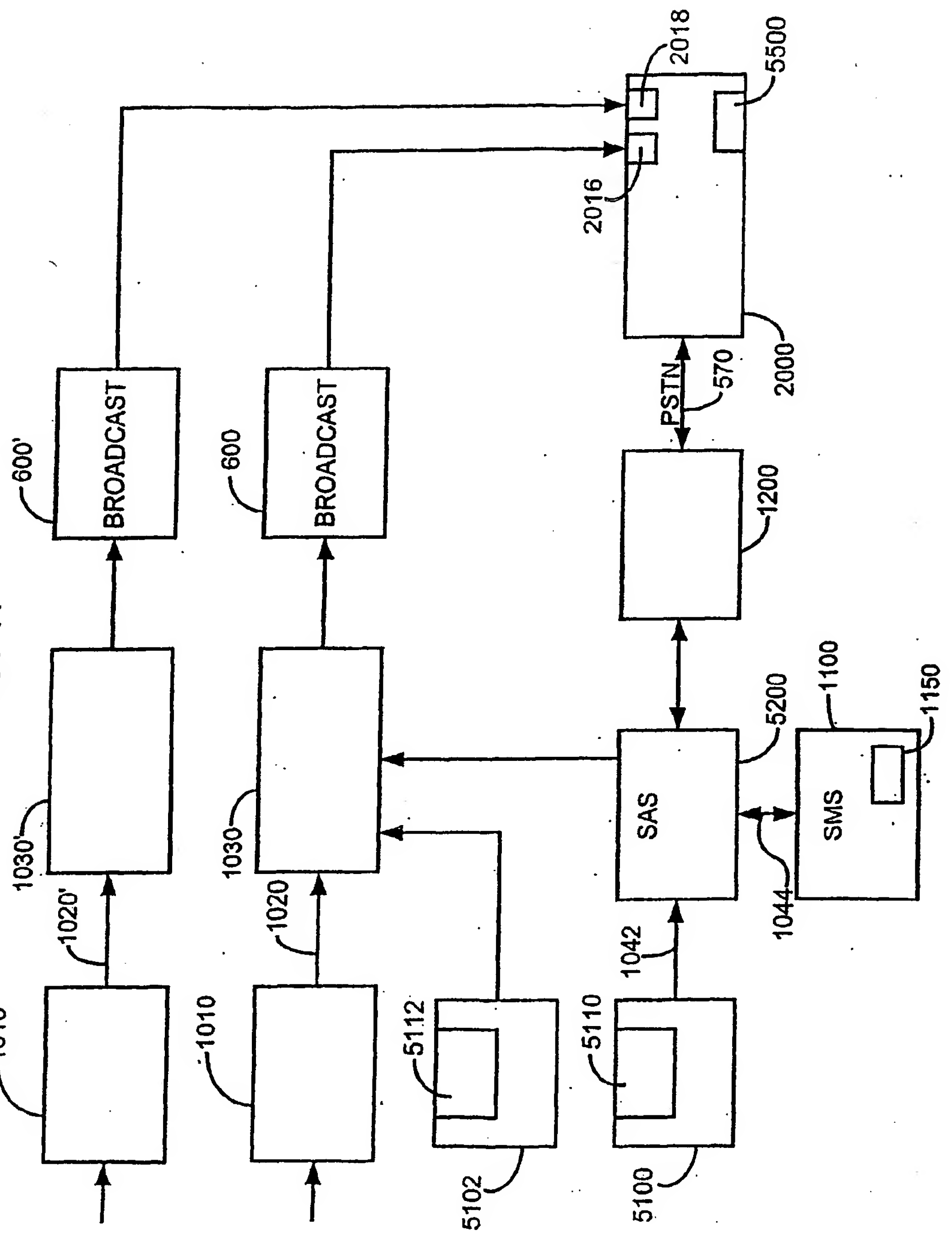

3146
Zapping Application
VIRTUAL MACHINE
3500
6402
CA KERNEL
ECMs
CWs
DLI
MLOAD
LCARD
CW
3760
3738
3740
6420
Descrambler
Transport Stream

FIG. 17


BROADCAST
BROADCAST
SAS
SMS
PSTN
600'
600
1010'
1010
1020'
1020
1030'
1030
5102
5112
5100
5110
1042
1044
5200
1100
1150
1200
570
2000
2016
2018
5500